Distributed Call Scheduling in Wireless Networks

Jean-Claude Bermond, Dorian Mazauric, Philippe Nain

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Distributed Call Scheduling in Wireless Networks

Jean-Claude Bermond — Dorian Mazauric — Philippe Nain

N° 6763

December 2008

Thème COM

ISSN 0249-6399 ISRN INRIA/RR--6763--FR+ENG

Distributed Call Scheduling in Wireless Networks

Jean-Claude Bermond*†, Dorian Mazauric*‡† , Philippe Nain*‡†

Thème COM — Systèmes communicants
Projets Mascotte and Maestro

Rapport de recherche n° 6763 — December 2008 — 27 pages

Abstract: We consider in this paper the distributed call scheduling problem in wireless networks. Due to interference in this kind of network, active links have to respect interference constraints. For example primary model permits to activate edges without common node. We assume we have random arrivals and the goal is to ensure for example the stability of the queueing system, bounding the average number of awaiting messages in the wireless network. Centralized algorithms permit to decide which links are activated at each step but it assumed a global knowledge of the network. It is necessary to design distributed algorithms which use local knowledge. We propose in this paper two distributed algorithms, valid for any binary interference model and with constant overhead, improving previous proposed ensuring only the first or the second feature.

Key-words: network, wireless, scheduling, distributed, algorithm, interference, stability.

This work was partially funded by Région PACA and by European project IST FET AEOLUS.

* MASCOTTE, INRIA, I3S, CNRS, Univ. Nice-Sophia Antipolis, Sophia Antipolis, France.
† firstname.lastname@sophia.inria.fr
‡ MAESTRO, INRIA, CNRS, LIRMM, Sophia Antipolis, France.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Algorithmes distributés d'ordonnancement dans les réseaux sans-fil

Résumé : Nous considérons dans cet article le problème d'ordonnancement distribué dans les réseaux sans-fil. En raison des interférences dans ce type de réseau, ne peuvent être activés simultanément que des liens n'interférant pas entre eux. Par exemple dans un modèle primaire, on ne peut activer que des liens deux à deux non adjacents. Nous nous plaçons dans un contexte d'arrivée aléatoire de messages et l'objectif est d'assurer un bon comportement du réseau en particulier d'assurer la stabilité des files d'attente, en limitant le nombre moyen de messages en attente. Des algorithmes centralisés permettant de décider quels liens sont activés à chaque étape existent mais ils supposent une connaissance globale du réseau et sont peu adaptés aux applications. Il est donc nécessaire de concevoir des algorithmes distribués qui utilisent une connaissance très locale du réseau. Nous proposons dans cet article deux algorithmes distribués, valides quelque soit le modèle d'interférence binaire et avec une phase de contrôle de durée constante, améliorant les algorithmes existants vérifiant uniquement l'un ou l'autre de ces deux critères.

Mots-clés : réseau, sans-fil, ordonnancement, distribué, algorithme, interférence, stabilité.

1 Introduction

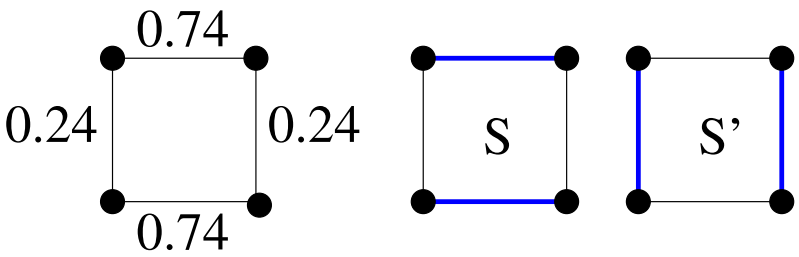
The call scheduling is a main problem in telecommunication networks. It was studied a lot for wire networks with the sufficient condition that the average arrival rate on each link of the network can not be greater than its capacity. For wireless networks (radio, ad hoc or sensor network for example), an additional constraint comes from interference problems. Thus during a step, it is not possible to activate some links of the network simultaneously because some interferences can not ensure good reception of messages. It is necessary before each sending phase to deal with the interference constraints and to choose a set of links in respect with the interference model. If the interference constraints are respected, then the different calls are guarantee to be good. Given a random traffic load for each link of the network, not necessary known by the designers, the goal is to send messages in the network scheduling the links in respect with the interference constraints. Some centralized algorithms have been proposed for this problem. In [**?**] [**?**], the authors designed centralized algorithms with stochastic arrivals. In [**?**] the proposed algorithms are valid for determistic arrivals. Pratically, the algorithms have to be distributed. For sensor networks for example, the decision has to be done locally, without exchanging messages with a central station in order to be energy sufficient for example. One of the most important thing is to deal with the interference constraints of wireless networks to design distributed algorithms for any binary interference model and with a constant number of control messages (constant overhead). The works presented in [**?**] [**?**] [**?**] described algorithms with an overhead increasing with the size of the network.

In this context, during a sending phase, the calls (activated links) have to respect the interference constraints. We will see (Section 2) how to model formally the interferences between communication links and we will describe our algorithms (Section 4). We consider synchronous networks and we divide time into steps, each of same size. During a step, a first control phase has been done to decide which links will be activated during the second phase, the sending phase. The goal is to activate links with an important number of awaiting messages. In a general case (interference model different than the primary node model), the problem of fnding a set of links respecting the interference constraints and maximizing the sum of the weights is *NP-Complete*. The proposed algorithms have to be distributed, valid for any binary interference model (Section 2) and with a constant number of control time slots. We assume that the traffic is single-hop, that is to say that a sent packet leaves the network immediately after even if we can adapt without difficulty our algorithms to take into account multi-hop traffic (Section 7). In this context the algorithms have to ensure that the average number of messages awaiting in each queue of the network is bounded.

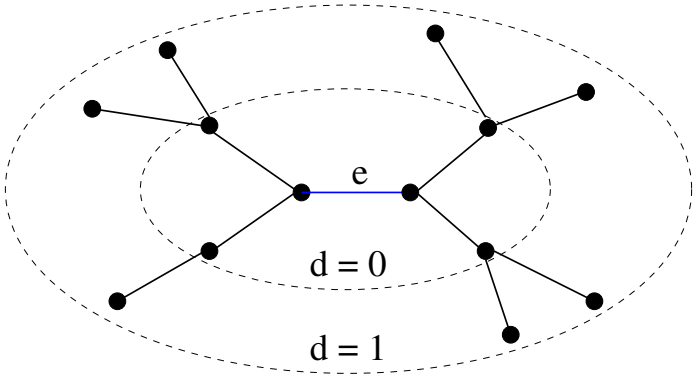
Figure 1(a) represents a wireless network of four stations modeling by a graph where each node represents one station and where edges represent the different possible calls, the different communication links. The primary node model allows to activate a set of links forming a matching in the associated graph (a matching is a set of edges without common node). We assume in this example that during a step, the activated links can be send a unique packet and the numbers located on links are the average arrival rate per step. Thus

a trivial scheduling is to choose three times over four the matching S and one time over four the matching S'. The average size of each queue is bounded.

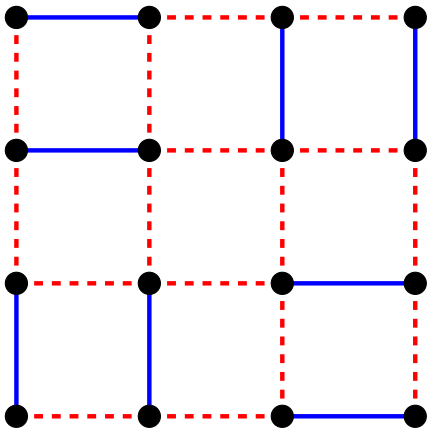
The rest of the paper is organized as follows. We present first the model used in our work (Section 2) before describing some previous works (Section 3). In Section 4, we proposed two distributed link scheduling algorithms in wireless networks in respect with the interference constraints. After we analyse these algorithms in Section 5 before describing some simulation results in Section 6. Finally we present in Section 7 generalizations of our algorithms to take into account multi-hop traffic and queues on nodes.



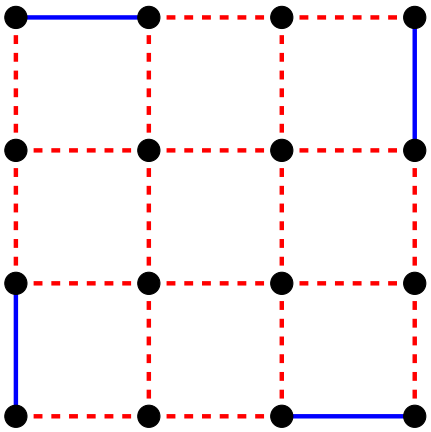
(a) Link Scheduling for a $4-cycle$ and $d=0$:
(S) three times over four and (S') one time over four.



(b) Examples of interference sets $\epsilon(e)$ for $d=0$ et $d=1$.



(c) Set of activated links (lines) for $d=0$.



(d) Set of activated links (lines) for $d=1$.

Figure 1: Examples of Link Scheduling (a), interference sets (b), and active links sets (c,d).

2 Modeling

A wireless network is represented by a non-oriented transmission graph $G = (V, E)$ where the nodes and the edges are respectively the stations and the communication links of the network. In other words, there exists an edge between $u \in V$ and $v \in V$ if and only if u (respectively v) can transmit a packet to v (respectively u). We assume that the links are symmetric (for example in radio networks we have acknowledgement messages). Interferences can be represented in different ways like signal-to-noise rate. This representation makes the analysis hard and so pratically we modeled it by binary models. Let us define formally the interference area for any edge of a transmission graph as follows:

Definition 1. *Given a transmission graph $G = (V, E)$, an edge $e \in E$ and an non negative integer d, the interference area of $e = u_1u_2$ is $\epsilon(e) = \{e' = v_1v_2 \in E, \exists i, j \in [1, 2], D(u_i, v_j) \leq d\}$. D is the distance in G, that is to say the length of one shortest path in G.*

The interference models will be defined for a fixed value of d. The particular case $d = 0$ represents the primary node model, it consists to find a matching in the transmission graph. Figure 1(a) represents the transmission graph of a wireless network with this primary model. In that case the two maximal macthings are S and S'. However the most realistic models are defined for $d > 0$. Figure 1(b) shows an example of interference areas for $d = 0$ and $d = 1$. Given a interference model (a fixed value of d), two active edges at step t can not be located in a same interference area. Considering synchronous networks, time is divided into steps, each of same size. One step contains two different phases: a *control phase* which consists to find a set of active links and a *sending phase*. We introduce for each edge of the transmission graph and for each step $t \geq 0$, the variable $a_t(e)$ such that $a_t(e) = 1$ if $e \in E$ is an active link, allowed to send data during step t ($a_t(e) = 0$ otherwise). Respecting the interference constraints, during a step, two active edges can not be in a same interference area. Formally $\forall e, e' \in E$ if $a_t(e) = a_t(e') = 1$ then $e' \notin \epsilon(e)$ and so $e \notin \epsilon(e')$. Figure 1(c) (respectively Figure 1(d)) describes an example of maximal set of active links (no edge can be added as active without change this set) for the interference model $d = 0$ (respectively $d = 1$). It corresponds tp a matching (respectively to an induced matching) in the transmission graph.

Traffic is dynamic and assumed single-hop, that is to say one packet sent through a communication link, leaves the network just after. It is possible to associated a packet to a link of the network, corresponding to an edge of the transmission graph. Thus a queue is defined for each link containing the packet which have to be send through this communication link. Designers will choose the policy of services depending of applications. We denote by $q_t(e)$ the number of messages in the queue of the link $e \in E$ at the beginning of step t. At each step, new packets arrive in the links (edges of the transmission graph) function of a probability law, specific to each link and not necessary known by designers. $A_t(e)$ is the number of messages arriving in edge $e \in E$ during step t. Remarks that these new packets will be really taken in consideration at step $t + 1$. Finally each communication link has a specific capacity, the number of messages which can be sent the link in one step (if the link is activated). We denote by $c(e)$ this capacity. Thus we have $q_{t+1}(e) = (q_t(e) - a_t(e).c(e))^+ + A_t(e)$ with $[x]^+ = max(x, 0)$.

3 Previous Works

We describe in this section two proposed algorithms of [**?**] and [**?**]. They are both distributed but the algorithm in [**?**] does not admit a constant overhead whereas the one described in [**?**] is valid only for the primary node model ($d = 0$).

3.1 Q-SCHED Algorithm

In [**?**], the authors presented a distributed algorithm for the scheduling problem in wireless networks. It is valid for any binary interference model ($\forall d \geq 0$) but the overhead increases with the size of the network. The network is assumed synchronous and each edge chooses a backoff value t_0 at the beginning of the control phase, that is to say here the date of its unique control message. If it receives a control message from one edge of its interference area, it will be inactive during the data phase. Otherwise it will be active. This simple algorithm is valid for any $d \geq 0$ but the choice of its backoff is function of the weights of the edges located in its interference area and function of the weights of edges in $\epsilon(e')$ with $e' \in \epsilon(e)$. An edge has to update at each step weights of edges located in its $2d - neighborhood$. Thus it is impossible to implement this algorithm pratically.

3.2 Augmenting Paths Algorithm

The algorithm described in [**?**] has a constant overhead but it is specific to the primary node model ($d = 0$). The goal is to find a new matching increasing the previous in terms of weights of active edges. Given a matching M_t at step t, during step $t + 1$ the algorithm will find paths, alternating edges in M_t and edge not in M_t. At the end, if the total weights of active edges of the alternating path is greater than the total weights, then the matching is changed, the status of each edge is changed. This control phase is constant but the algorithm is designed specifically for primary node model, the less realistic. Furthermore the convergence is very slow (see Section 6).

Finally it is necessary to design distributed (local) algorithms with constant number of time slots (constant overhead), valid for any binary interference model ($\forall d \geq 0$) and with fast convergences.

4 Our Algorithms

We propose in this section two distributed link scheduling algorithms for wireless networks, valids for any binary interference model and ensuring a constant overhead. As previously described a network is modeled by a valued transmission graph $G = (V, E)$. We will present our algorithms for the case which values $q_t(e)$ are in the finite set $[0, K-1]$ where K depends on the number of available number of control time slots. The general case can be reduced to this one partitioning $q_t(e)$ in K classes. For example we can put the $q_t(e) \geq M$ (M a constant) in the class K partitioning others in $K-1$ classes of almost same size. We assume in the rest of the paper that $\forall t \geq 0, \forall e \in E$, $q_t(e) \in [0, K-1]$.

4.1 Log Algorithm 1

As defined before, we consider synchronous networks and in this context, time is divided into steps, each of same size. A step contains two phases: a control phase and a data phase. We will describe precisely the control phase of Log Algorithm 1 which is divided into two subphases: the *regular subphase* with T time slots and the *random subphase*. We will first describe the regular before motivating the need of the additional subphase. Let us define three possible status for an edge during the control phase:

Definition 2. *Given a transmission graph $G = (V, E)$, an edge $e \in E$ can be:*
- active, e will send messages during the data phase;
- inactive, e will not send data;
- undetermined if the choice is not already done.

At the beginning of each control phase, each edge is undetermined and at the end each is either active or inactive. Describe now the protocol of an edge $e \in E$ already undetermined during a time slot i, $1 \leq i \leq T$:
a) if e does not send control message and does not receive one from an edge of its interference area $\epsilon(e)$, then e becomes inactive;
b) if e sends one control message and does not receive one from an edge of its interference area $\epsilon(e)$, then e becomes active;
c) otherwise e remains undetermined.

After control time slot T, each undetermined edge becomes inactive. Explain now the protocol used by edges to send control messages. Each edge $e \in E$ computes a *control vector* $v_{e,t} = (v_{e,t}(1), v_{e,t}(2), ..., v_{e,t}(T))$ where $v_{e,t}(i) = 1$ or 0. If $v_{e,t}(i) = 1$, edge e will send a control message if it is already undetermined, otherwise e does not send one. $v_{e,t}(i)$ ($i = 1..T-1$) correspond to the ith bit of its weight written in a binary way, adding eventually some additional zeros. Remark that edges with odd weights send one control message at time slot $T-1$. To be fair, time slot T is for even weight edges already undetermined: $v_{e,t}(T) = 1$ if $q_t(e)$ is even, $v_{e,t}(T) = 0$ otherwise. As explained before, weights are bounded by K, and so we can write for each possible weight the sending protocol used by edges. For

Table 1: Control vector $v_{t,e}$ of edge $e \in E$ at step $t \geq 0$ for each possible weight $q_t(e)$ for Log Algorithm 1 ($K = 16$).

$q_t(e)$	$v_{t,e}(1)$	$v_{t,e}(2)$	$v_{t,e}(3)$	$v_{t,e}(4)$	$v_{t,e}(5)$
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	0

example in Table 1, the control vector is described for each possible weight q_1 if $K = 16$ (and so $T = 5$).

Figure 2 represents the regular control subphase of Log Algorithm 1 for a 4×4 grid, with $d = 0$ and $K = 16$. In this example the matching ($d = 0$) is maximal, that is to say no edge can be added to the current matching without changing it. In Section 9, we have an example of random transmission graph which the regular subphase of Log Algorithm 1 is applied for a more realistic interference model $d = 1$. However in the example of Figure, the regular subphase of Log Algorithm 1 is applied on a path and the edge of weight 15 is the unique active edge. To deal with the non maximal set of active edges after the regular control subphase, we add a constant additional number of control time slots (corresponding to the random subphase) in order to have active edges set with greater cardinality. Thus the inactive edges without active in its interference area become again undetermined (Figure 3(f)). It is possible with one control time slot. After that each undetermined edge chooses a random backoff value. In the example 2 extra time slots are used, thus the edges choose 1 or 2. The protocol is the same than before and in Figure 3(h), the two edges of backoff 1 send a control message and become active, the other edges receive one message and so become inactive. Pratically 3 additional random time slots are sufficient but a better solution is in the next proposed algorithm: Log Algorithm 2.

Lemma 3. *The number of control time slots of Log Algorithm 1 is* $T_1 = \lceil log_2(K) \rceil + 4 = O(log_2(K))$.

Proof. The number of control time slots is the size of the control vector plus the number of additional time slots. The control vector is the number of bits of the maximum weight written in a binary way. Thus $T_1 = \lceil log_2(K) \rceil + 4 = O(log_2(K))$. □

4.2 Log Algorithm 2

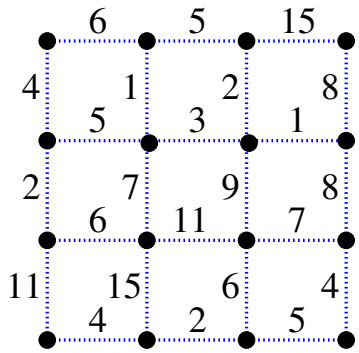
Log Algorithm 2 is directly adapted from Log Algorithm 1 permitting to deal with the problem of non maximal active edges set in a better way. The idea is to repeat the regular subphase of Log Algorithm 1. As described before Figure 3(a-e) represents a 10-nodes path in which the regular subphase of Log Algorithm 1 activates only the link of weight 15 and so the matching ($d = 0$ in this example) is not maximal. Using 1 control time slot, each inactive edge without active in its interference area, becomes undetermined (Figure 4(b)). After repeating regular suphase one time more, we get the matching on Figure 4(c). In the same way, the regular subphase is applied one more time and we get a maximal matching (Figure 4(e)).

Another idea is to define new (virtual) weights $q'_t(e)$ for each edge $e \in E$ such that $q'_t(e) = c_2.q_t(e)) + g(t, e)$ with $0 \leq g(t, e) < c_2$ and c_2 a constant. More precisely $g(t+t_0, e) = g(t, e) + mod(t_0, c_2)$ with $g(0, e)$ a constant ($0 \leq g(0, e) < c_2$). Taking in consideration these new weights, we repeat n times the regular subphase of Log Algorithm 1 (n will be defined later). Assuming that $q_t(e) < K$, we have $q'_t(e) < K' = c_2K$. Table 2 represents control vectors for each possible weight with $K = 8$ and $c_2 = 2$ (and so $K' = 16$). It can be correspond to a path and the primary node model $d = 0$ such that $\forall e \in E$ and $\forall e' \in \epsilon(e)$, then $g(0, e) \neq g(0, e')$. In that case, it is sure that two edges belonging to a same interference area can be compared (virtual weights are different). We will in Section 5 that for each edge $e \in E$, then there exists an edge $e' \in \epsilon(e) \cup \{e\}$ such that $a_t(e') = 1$. Finally it is possible to remark that this transformation does not change the weights order: $\forall e, e' \in E$, if $q_t(e) \leq q_t(e')$, then $q'_t(e) \leq q'_t(e')$. From Lemma 7 in Section 5, $n = Log_2(K')$.

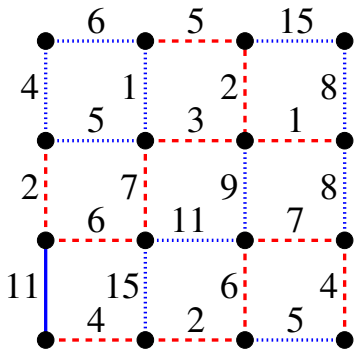
Lemma 4. *The number of control time slots of Log Algorithm 2 is* $T_2 = O(log_2(K)^2)$.

Proof. From Lemma 3 we know that the regular subphase of Log Alorithm 1 with new weights $q'_t(e)$ requires $\lceil log_2(K') \rceil + 1$ control time slots. Repeating it n times we have $T_2 = n(\lceil log_2(K') \rceil + 1)$ and from Lemma 7 we have $T_2 = \lceil log_2(K') \rceil (\lceil log_2(K') \rceil + 1)$. Finally since $K' = c_2.K$ with c_2 a constant, we have $T_2 = O(log_2(K)^2)$. □

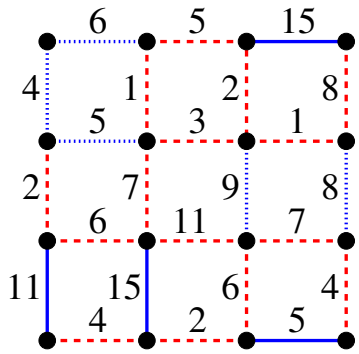
Figure 5 describes a critical example of transmission graph with almost similar weights. Using the transformation described before (Figure 5(b)) shows new weights of edges. Since the interference model is the primary node and the transmission graph is a path, $c_2 = 2$ is sufficient in order to have the property that two edges in a same interference area do not have same weight. Figure 5(h) shows the active links set. In next section, it is interesting to analyse these algorithms, proving their stabilities for example.



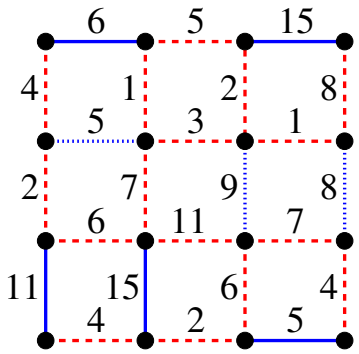
(a) Initial Graph



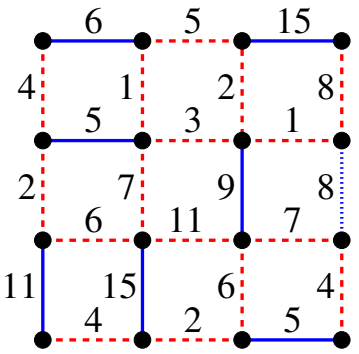
(b) $i = 1$



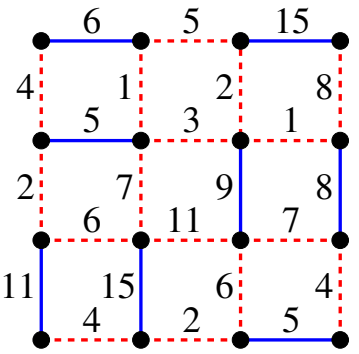
(c) $i = 2$



(d) $i = 3$



(e) $i = 4$



(f) $i = 5$

Figure 2: Regular subphase of Log Algorithm 1 for a 4×4 grid, $d = 0$ and $K = 16$ (5 control time slots). Blue edges represent active, red represent inactives and dotted-lines are already undetermined.

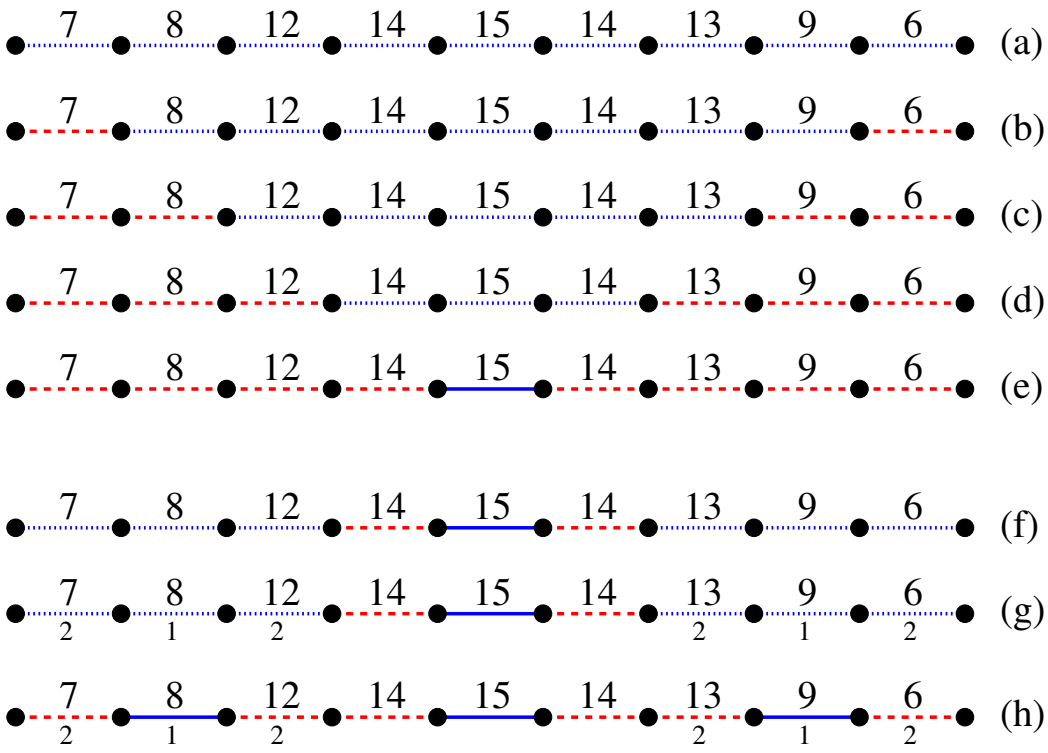


Figure 3: Control phase of Log Algorithm 1 for a 10-nodes path with $d = 0$ and $K = 16$:
- regular subphase (a-e);
- random subphase (f-h).

Table 2: Control vector $v_{t,e}$ of edge $e \in E$ at step $t \geq 0$ for each possible weight $q_t(e)$ for Log Algorithm 2 ($K = 8$, $c_2 = 2$, $K' = 2K = 16$).

$q_t(e)$	$g(t,e)$	$q'_t(e)$	$v_{t,e}(1)$	$v_{t,e}(2)$	$v_{t,e}(3)$	$v_{t,e}(4)$	$v_{t,e}(5)$
0	0	0	0	0	0	0	1
0	1	1	0	0	0	1	0
1	0	2	0	0	1	0	1
1	1	3	0	0	1	1	0
2	0	4	0	1	0	0	1
2	1	5	0	1	0	1	0
3	0	6	0	1	1	0	1
3	1	7	0	1	1	1	0
4	0	8	1	0	0	0	1
4	1	9	1	0	0	1	0
5	0	10	1	0	1	0	1
5	1	11	1	0	1	1	0
6	0	12	1	1	0	0	1
6	1	13	1	1	0	1	0
7	0	14	1	1	1	0	1
7	1	15	1	1	1	1	0

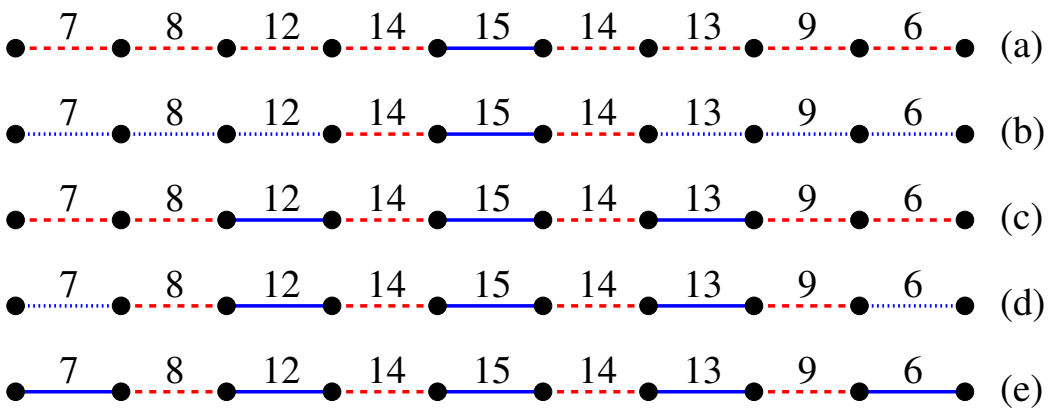


Figure 4: Log Algorithm 2 for a 10-nodes path 1 with $d = 0$ and $K = 16$.

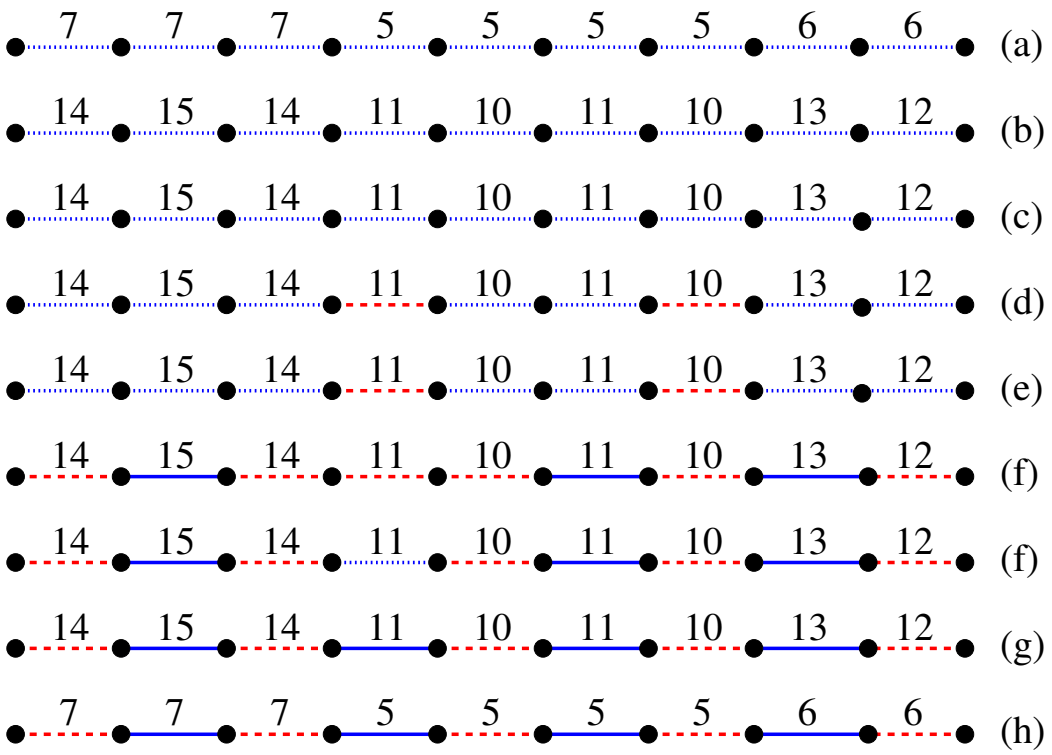


Figure 5: Log Algorithm 2 for a 10-nodes path, $d = 0$, $K = 8$ and $c_2 = 2$.

5 Analysis

In this section, we analyse our algorithms in order to describe some interesting properties and to get sufficient conditions for stability.

Definition 5. *Given a valuated transmission graph $G = (V, E)$ with weights $q_t(e)$ for each edge $e \in E$, we say that $e \in E$ is a strict local maximum at step $t \geq 0$ if and only if $\forall e' \in \epsilon(e), q_t(e') < q_t(e)$ or $q'_t(e') < q'_t(e)$.*

Lemma 6. *A strict local maximum edge is always active with Log Algorithm i (i=1,2).*

Proof. From our algorithm designs, an edge is inactive if it receives a message during a control time slot from an edge of its interference area without sending one itself or if it is undetermined after the last control time slot. In that case the edge can not be a strict local maxium edge and we have a contradiction. Thus a maximum is necessarily active. □

Lemma 7. *Let $G = (V, E)$ be a valuated graph with $q_t(e)$ the weight of each edge $e \in E$ at step $t \geq 0$ such that if $e' \in \epsilon(e)$, then $q_t(e) \neq q_t(e')$. After the regular subphase of Log Algorithm 1 at step t, the maximum diameter among subgraphs induced by inactive edges in G is at most $log_2(K)$.*

Proof. Without lose of generality, one worst case can be represented by a path with edges $e_{-n}, e_{-(n-1)}, ..., e_{-1}, e_0, e_1, ..., e_{n-1}, e_n$ such that:
$\forall i \in [-(n-1); n-1]$, we have $v_{e_{-n}}(1) = v_{e_n}(1) = 0 = 1 - v_{e_i}(1)$ and;
$\forall i \in [-(n-2); n-2]$, we have $v_{e_{-(n-1)}}(2) = v_{e_{n-1}}(2) = 0 = 1 - v_{e_i}(2)$ and;
...
$\forall i \in [-(n-j); n-j]$, we have $v_{e_{-(n-j)}}(j) = v_{e_{n-j}}(j) = 0 = 1 - v_{e_i}(2)$ and;
...
$v_{e_{-2}}(n-1) = v_{e_2}(n-1) = 0 = 1 - v_{e_{-1}}(n-1) = 1 - v_{e_0}(n-1) = 1 - v_{e_1}(n-1)$ and;
$v_{e_{-1}}(n) = v_{e_1}(n) = 0 = 1 - v_{e_0}(n)$.
In that case e_0 is the unique active edge and to maximize the diameter it is necessary to take $n = log_2(K)$. Thus the maximum diameter among the induced subgraphs is at most $log_2(K)$. □

Lemma 8. *Considering Log Algorithm 2 and given a transmission graph $G = (V, E)$, $\forall e \in E, \forall t \geq 0$, then $\exists e' \in \epsilon(e) \cup \{e\}$ such that $a_t(e') = 1$.*

Proof. First we can transform the weight of each edge $e \in E$ as previously described ensuring that $\forall e \in E, \forall e' \in \epsilon(e)$, we have $q'_t(e) \neq q'_t(e')$. Thus from Lemma 7 and by induction on the maximum diameter among the subgraphs induced by inactive edges, it is obvious that after a control phase of Log Algorithm 2 at step t, $\forall e \in E, \exists e' \in \epsilon(e) \cup \{e\}$ such that $a_t(e') = 1$. □

Corollary 9. *For each $e \in E$, we define $a_t(\epsilon(e))$ the random variable counting the number of active edges in the interference area of e at step t. From Lemma 8, we have: $\forall e \in E$, $E[a_t(\epsilon(e))] \geq 1$ with Log Algorithm 2.*

Notation 1. *Any vector* $\mathbf{y}$ *in* N^N *is represented by* $\mathbf{y} = (y(1), \ldots, y(N))$. *In particular,* $\mathbf{y}_t := (y_t(1), \ldots, y_t(N))$. *We define* $|\mathbf{y}| = \sum_{e=1}^{N} y(e)$.

Let $c(e) > 0$ *be the (constant) capacity of link* $e \in E$, *defined as the maximum number of packets that can be served in a time-slot.*

For each $e \in E$ *we assume that the rvs* $\{A_t(e), t \geq 1\}$ *are iid; further the sequences* $\{A_t(e), t \geq 1\}$ *and* $\{A_t(e'), t \geq 1\}$ *are mutually independent for all* $e \in E$ *and* $e' \in E$ *such that* $e \neq e'$.

Recall that $q_t(e)$ *be the backlog on link* $e \in E$ *at the beginning of slot* t. *We have (with* $[x]^+ = \max(x, 0)$*)*

$$q_{t+1}(e) = [q_t(e) - a_t(e)c(e)]^+ + A_t(e), \quad e = 1, \ldots, N, \tag{1}$$

where $a_t(e) = 1$ *(resp.* $a_t(e) = 0$*) if link* $e \in E$ *is active (resp. inactive) in slot* t.

Definition 10 (Matching)**.**
A (stationary) matching is a family of probability distributions $\{I(.; \mathbf{q}), \mathbf{q} \in \mathbb{N}^N\}$ *on* $\{0, 1\}^N$ *with the interpretation that*

$$P(\mathbf{a}_t = \mathbf{a} \,|\, \mathbf{q}_t = \mathbf{q}) = I(\mathbf{a}; \mathbf{q}) \tag{2}$$

for all $\mathbf{q} \in \mathbb{N}^N$, $t \geq 1$. *The matching is stationary in the sense that the right-hand side of (2) does not depend on* t.

Under the above assumption, for any matching, $\{\mathbf{q}_t, t \geq 1\}$ is a homogeneous discrete-time Markov chain (MC) on the state space $\mathbb{N}^N$.

Assumption 1. *(a)* $\overline{A}(e) := E[A_t(e)] < \infty$, $\overline{A^2}(e) := E[A_t(e)^2] < \infty$.

(b) For each $e = 1, \ldots, N$

$$\frac{\overline{A}(e)}{c(e)} < G(e) := \lim_{K \to \infty} \inf_{|\mathbf{q}| \geq K} \sum_{\substack{a(e)=1 \\ \mathbf{a} \in \{0,1\}^N}} I(\mathbf{a}; \mathbf{q}). \tag{3}$$

The existence of the limit in (3) is ensured by the fact that $\inf_{|\mathbf{q}| \geq K} \sum_{\substack{a(e)=1 \\ \mathbf{a} \in \{0,1\}^N}} I(\mathbf{a}; \mathbf{q})$ *is non-decreasing in* K.

The following result holds.

Theorem 11. *Stability of the MC* $\{\mathbf{q}_t, t \geq 1\}$
Consider a matching $\{I(.; \mathbf{q}), \mathbf{q} \in \mathbb{N}^N\}$ *such that the MC* $\{\mathbf{q}_t, t \geq 1\}$ *is irreducible. Then,* $\{\mathbf{q}_t, t \geq 1\}$ *is stable under assumption* **A1**.

Proof. We will show that Foster's criterion applies, namely, there exists a (Lyapounov) function $V : \mathbb{N}^N \to [0,\infty)$, a finite set $\mathcal{F} \subset \mathbb{N}^N$ and a constant $\delta > 0$ such that (i) $\Delta(\mathbf{q}) := E[V(\mathbf{q}_{t+1}) - V(\mathbf{q}_t) \,|\, \mathbf{q}_t = \mathbf{q}] < -\delta$ for all $\mathbf{q} \in \mathcal{F}^c$ and (ii) $\sup_{\mathbf{q}\in\mathcal{F}} \Delta(\mathbf{q}) < \infty$.

Take $V(\mathbf{q}) = \sum_{e=1}^N q(e)^2$. This is a Lyapounov function.

We have from (1) (Hint: use $([x]^+)^2 \le x^2$ and $\max(x-y,0) \le x$ for $x, y \ge 0$)

$$\begin{aligned}
\Delta(\mathbf{q}) &= E\left[\sum_{e=1}^N \left([q(e) - a_t(e)c(e)]^+ + A_t(e)\right)^2 - \sum_{e=1}^N q(e)^2 \,|\, \mathbf{q}_t = \mathbf{q}\right] \qquad (4)\\
&= E\left[\sum_{e=1}^N \left(([q(e) - a_t(e)c(e)]^+)^2 + 2A_t(e)[q(e)) - a_t(e)c(e)]^+ + A_t(e)^2 - q(e)^2\right) \,|\, \mathbf{q}_t = \mathbf{q}\right]\\
&\le E\left[\sum_{e=1}^N \left((q(e) - a_t(e)c(e))^2 + 2A_t(e)q(e) + A_t(e)^2 - q(e)^2\right) \,|\, \mathbf{q}_t = \mathbf{q}\right]\\
&= \sum_{e=1}^N \Big(2q(e)\left(\overline{A}(e) - c(e)E[a_t(e) \,|\, \mathbf{q}_t = \mathbf{q}]\right) + c(e)^2 E[a_t(e)^2 \,|\, \mathbf{q}_t = \mathbf{q}] + \overline{A^2}(e)\Big)\\
&\le 2\sum_{e=1}^N q(e)\left(\overline{A}(e) - c(e)E[a_t(e) \,|\, \mathbf{q}_t = \mathbf{q}]\right) + \sum_{e=1}^N (\overline{A^2}(e) + c(e)^2) \qquad (5)
\end{aligned}$$

where we have used the fact that $0 \le a_t(e) \le 1$ to derive the last inequality.

From the equality $E[a_t(e) \,|\, \mathbf{q}_t = \mathbf{q}] = P(a_t(e) = 1 \,|\, \mathbf{q}_t = \mathbf{q}]$ (which holds since $a_t(e) \in \{0,1\}$) and (2) we see that

$$E[a_t(e) \,|\, \mathbf{q}_t = \mathbf{q}] = \sum_{\substack{a_t(e)=1\\ \mathbf{a}\in\{0,1\}^N}} I(\mathbf{a};\mathbf{q}). \qquad (6)$$

Combining (5) and (6) gives

$$\Delta(\mathbf{q}) \le 2\sum_{e=1}^N q(e)\left(\overline{A}(e) - c(e)\sum_{\substack{a_t(e)=1\\ \mathbf{a}\in\{0,1\}^N}} I(\mathbf{a};\mathbf{q})\right) + \sum_{e=1}^N (\overline{A^2}(e) + c(e)^2) \qquad (7)$$

Take $0 < \epsilon < \min_{e=1,\ldots,N}(G(e) - \overline{A(e)}/c(e))$. The existence of ϵ is ensured by Assumption **A1**(a). From now on ϵ is fixed.

By definition of a limit we see from (3) that for each $e = 1, \ldots, N$ there exists an integer $K(e)$ such that for all $K \ge K(e)$,

$$G(e) - \epsilon < \inf_{|\mathbf{q}|\ge K} \sum_{\substack{a_t(e)=1\\ \mathbf{a}\in\{0,1\}^N}} I(\mathbf{a};\mathbf{q}) < G(e) + \epsilon$$

so that, from the choice of ϵ,

$$\frac{\overline{A}(e)}{c(e)} < \inf_{|\mathbf{q}|\geq K} \sum_{\substack{a_t(e)=1 \\ \mathbf{a}\in\{0,1\}^N}} I(\mathbf{a};\mathbf{q}), \quad e=1,\ldots,N, \tag{8}$$

for all $K \geq K^\star$ where $K^\star := \max\{K(1),\ldots,K(N)\} < \infty$.

For $K \geq K^\star$ define

$$\rho := \max_{e=1,\ldots,N} \left\{ \overline{A}(e) - c(e) \inf_{|\mathbf{q}|\geq K} \sum_{\substack{a_t(e)=1 \\ \mathbf{a}\in\{0,1\}^N}} I(\mathbf{a};\mathbf{q}) \right\}$$

Obverve that $\rho < 0$ from (8).

Let $\delta > 0$ be an arbitrary constant. For $|\mathbf{q}| \geq K^\star$ we have from (7) and the definition of ρ

$$\Delta(\mathbf{q}) \leq 2\rho \sum_{e=1}^{N} q(e) + \sum_{e=1}^{N} (\overline{A^2}(e) + c(e)^2) \tag{9}$$

so that $\Delta(\mathbf{q}) < -\delta$ as long as

$$|\mathbf{q}| > \Gamma := \max \left\{ K^\star, \frac{\sum_{e=1}^{N} (\overline{A^2}(e) + c(e)^2) + \delta}{-2\rho} \right\}. \tag{10}$$

The finiteness of $\overline{A^2}(e)$ for each e (see Assumption A1(a)) ensures the finiteness of Γ.

Define the set $\mathcal{F} = \{\mathbf{q} \in \mathbb{N}^N : |\mathbf{q}| \leq M\}$ where M is any finite integer larger than Γ.

Eqns (9)-(10) show that part (i) of Foster's criterion applies for all $\mathbf{q} \in \mathcal{F}^c$. Part (ii) applies too since we find from (4) that $\Delta(\mathbf{q}) \leq 2M \sum_{e=1}^{M} \overline{A}(e) + \sum_{e=1}^{M} \overline{A^2}(e) < \infty$ for all $\mathbf{q} \in \mathcal{F}$, which concludes the proof.

□

An *admissible* matching is a matching which satisfies Definition 10 and meets the interference constraints.

To do: ca serait bien de trouver une borne inférieure pour la partie droite de (3), du moins dans le cas de certaines topologies.

6 Simulation Results

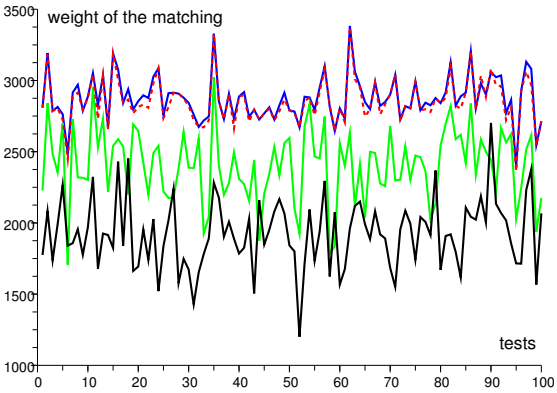
We present here some simulation results. We have implemented some classical topologies like path and grid, and also random graphs representing real wireless networks (see an example of random transmission graph in Section 9). We have implemented centralized algorithms and some distributed like Augmenting Path Algorithm [**?**], Log Algorithms described in Section 4 and one fully random algorithm. The results are presented for the binary interference model $d = 0$.

6.1 One Step Efficiency

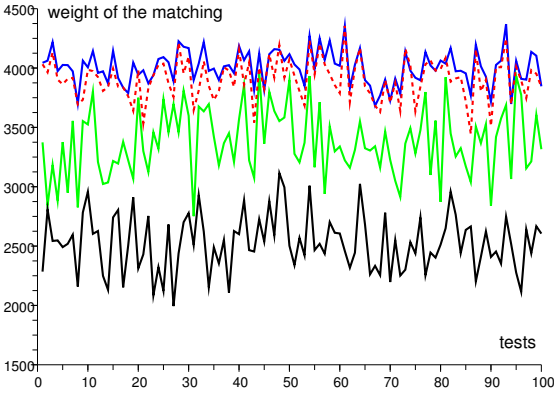
We describe here the efficiencies of different algorithms for a single step (one control phase and one data phase) given a particular distribution of the weights (sizes of queues). We compare the active weights for the different algorithms. Figures 6(a) and 6(b) represent the weight of the mathcing for each of the algorithm for 100 tests with random uniform weights, respectively for a 100-edges path and for a 10×10grid. The regular subphase of Log Algorithm 1 gives a total weight almost equal to the one given for the centralized algorithm (Figure 6(a)). With 3 additional random time slots Log Algorithm 1 gives also a total weight almost equal for the grid (Figure 6(b)). Finally Log Algorithm 2 gives the same weight than the weight of the centralized algorithm.

6.2 Stability Results

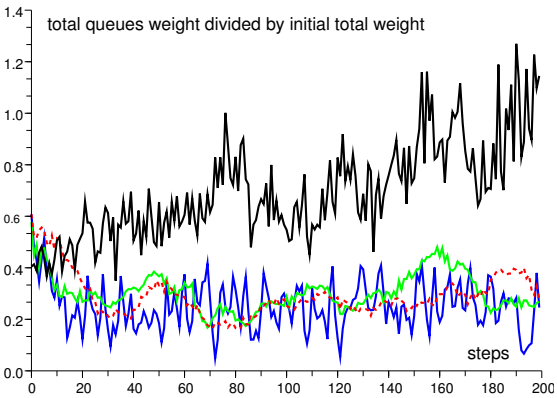
We study here the stability of the queueing system of the network simuling consecutive steps with packet arrivals. Figure 6(c) shows the stability of Log Algorithms and the centralized algorithm for a 100-edges path with a capacity $c(e) = 30$ for each link e, an average arrival number equals to 12 (uniformily between 0 and 24) and 200 steps. The random algorithm is not stable. Finally simulation results give an approximation of $G(e)$ (Theorem 11).



(a) Weight of the matching at one step for a 100-edges path, uniform random weights, $d = 0$ et 100 tests.



(b) Weight of the matching at one step for a 10×10 grid, uniform random weights, $d = 0$ et 100 tests.



(c) Total current weight over intial toal weight for a 100-edges path, $\forall e\ \overline{A(e)} = 12$, $c(e) = 30$ and 200 steps.

Figure 6: Simulation results: centralized algorithm (blue), Log Algorithm 1 (green), Log Algorithm 2 (dotted red) and one random algorithm (black).

7 Discussion

In this section we describe some possible generalization and futur works such that dealing with multi-hop traffic and queues located on nodes.

7.1 Multi-hop traffic

We can adapt Log Algorithms to take into consideration multi-hop traffic. From Log Algorithm 1 (respectively Log Algorithm 2) we can change previous weights (respectively virtual weights) into multi-hop weights $q_t^m(e)$ (respectively virtual multi-hop weights $q_t^{m\prime}(e)$). More precisely $q_t^m(e) = \sum p_t^i(e).h_t^i(e)$ where $p_t^i(e)$ is the packet numbered i in the queue of link e at step t and $h_t^i(e)$ is the number of remaining hops of the previous packet. Furthermore $q_t^{m\prime}(e) = c_2.q_t^m(e) + g(t,e)$. We assumed here that the routing is pre-computed. The policy of services can changed with application requirements.

7.2 Queues on nodes

We can also adapt Log Algorithm to weights on nodes instead of weights on links (edges). To do it, it is possible to get a similar algorithm with an overhead $T' = 3.T_2$ (recall that T_2 is the number of control time slots in Log Algorithm 2). Thus the overhead remains constant. We can explain easily the new version of Log Algorithm with weights on nodes using an example where $d = 0$. Fig 7 represents a *9-nodes path* where almost each node has 2 different weights corresponding to its 2 queues (except of course the left and right node). One version of new Log Algorithm is that each node chooses the queue maximum length (see Fig 7 (b)). After that each node computes its vector (the same computation than the previous Log Algorithm). At step (c), each node sends a message and so all nodes remains undetermined. At step (d), the nodes of weight 6 or 7 send a message. The right node of weight 6 has sent a message to 5 and 4 with the identification number of the node of weight 4. This one sends a message to 6 to confirm its request and 6 sends again a message to confirm. Thus node 6 actives the link (Fig7 (e)). For the other nodes, there is a problem of interference and so only inactive some nodes (of weight 4 and 5). In the same way, two other links are activated (Fig 7 (f,g,h,i)). As for Log Algorithm 2, this protocol is repeated and at the end there are 4 activated links (Fig 7 (j,k)).

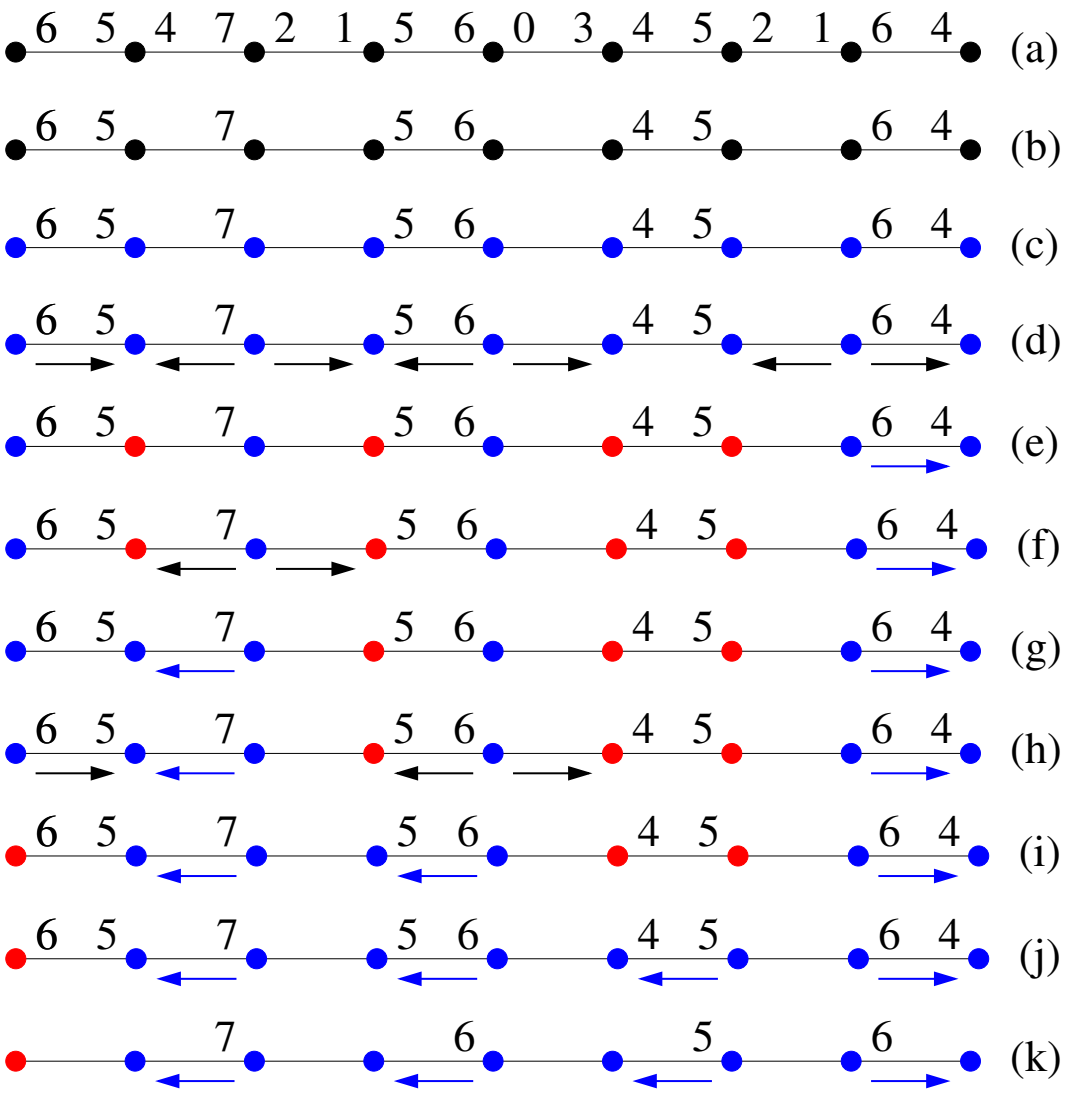


Figure 7: Generalization of Log Algorithm for a 9-nodes path, $d = 0$ and $K = 8$.

8 Conclusion

We proposed in this article two distributed scheduling algorithms for wireless networks, valid for any binary interference model and with a constant number of control time slots. It will be interesting to determine the value of $G(e)$ of Theorem 11. Finally we can adapt our algorithms to queues located on nodes and also taking in consideration multi-hop traffic.

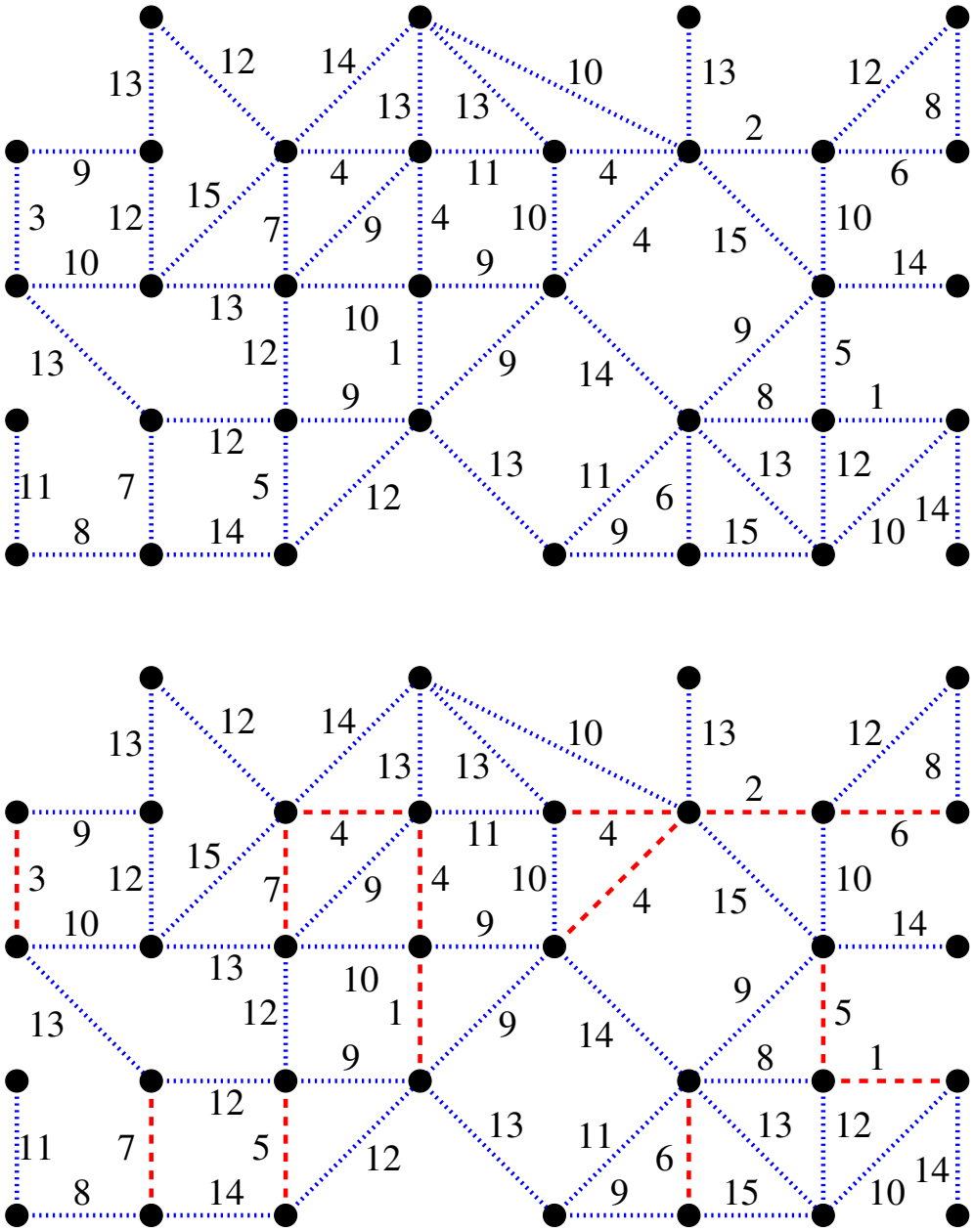
Acknowledgments

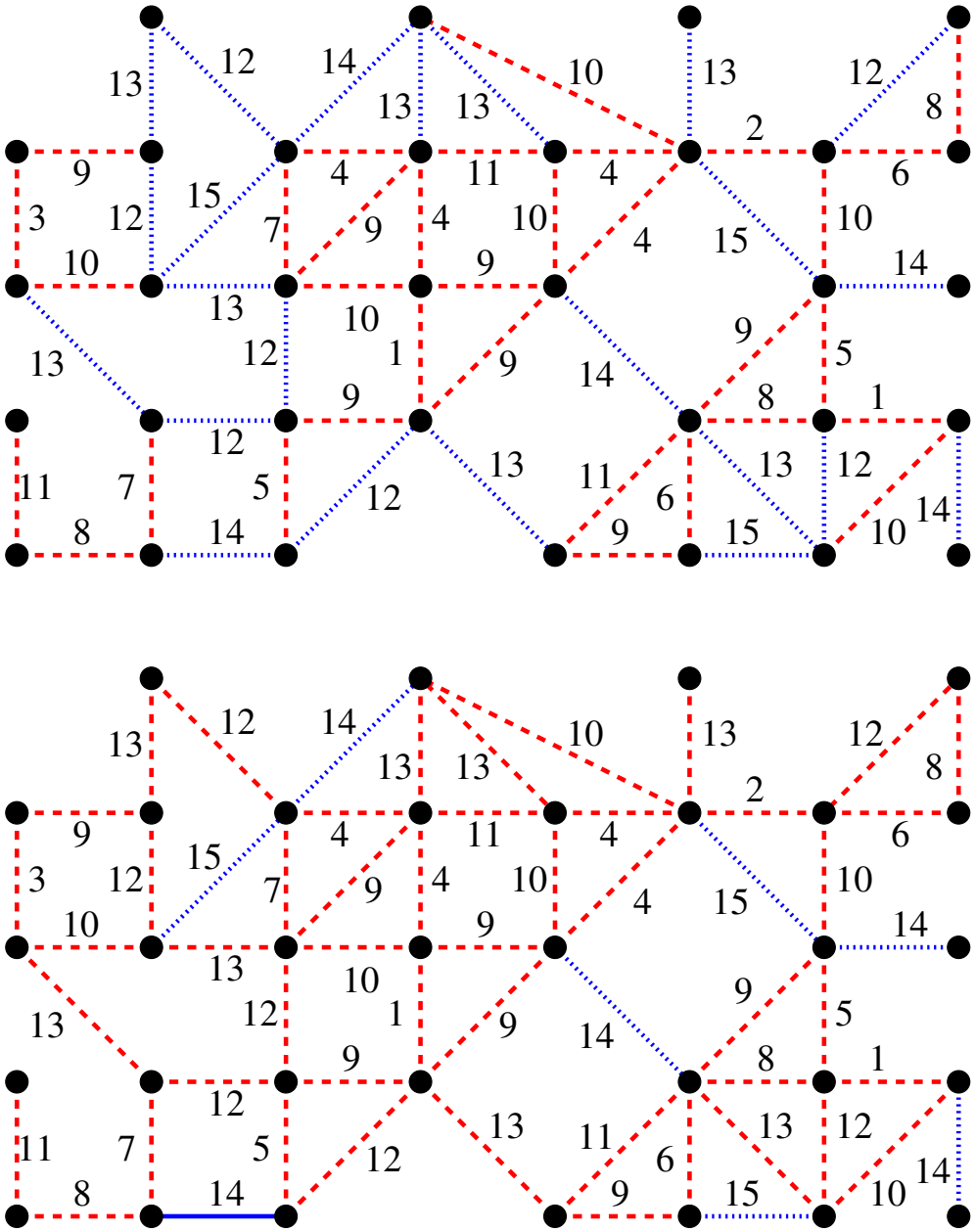
This work has been partially supported by région PACA, and European project IST FET AEOLUS.

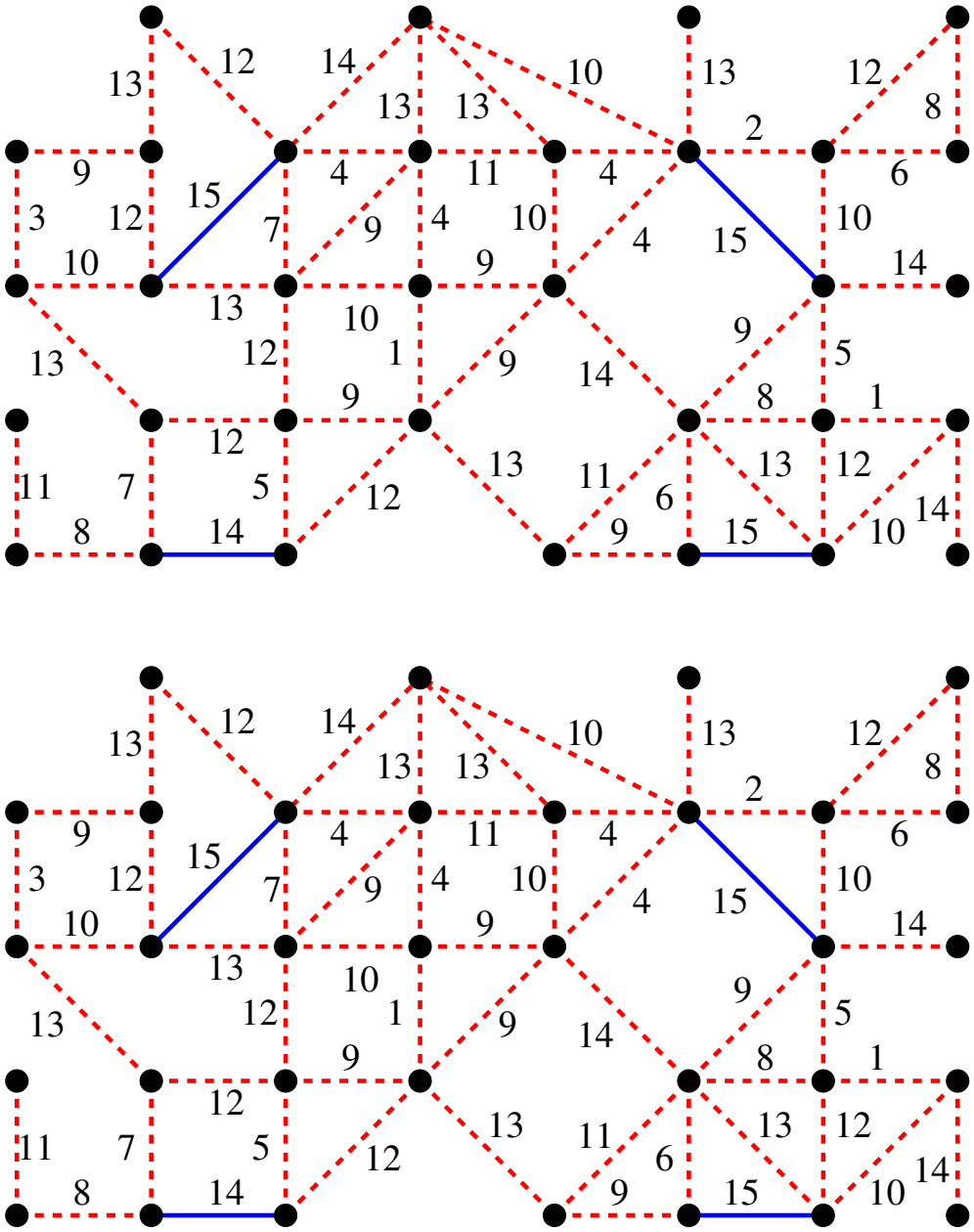
9 Appendix

We can see (in the above figure) the regular subphase of Log Algorithm 1 for a random transmission graph $G = (V, E)$ with the interference model $d = 1$.

- **(a)** The first subfigure represents the transmission graph before the control subphase of Log Algorithm 1 of step t. Each edge is undetermined, represented in G by blue dotted-lines.
- **(b)** Each edge $e \in E$ such that $v_{1,e} = 1$ sends a control message, that is to say each edge with weight in the finite set $[8, 15]$. In this example, each of these edges stays undetermined because it receives at least one control message. All others (with weight in the set $[0, 7]$ become inactive (red).
- **(c)** Undetermined edges $e \in e$ such that $v_{2,e} = 1$ send a control message. It corresponds to edges with weight in $[12, 15]$.
- **(d)** Before the third control time slot, edges with weight 14 or 15 are the unique undetermined edges. Others are inactives. After this time slot, we have one active, the edge 14.
- **(e)** After time slot 4, we have a maximal set of active edges corresponding to the interference model $d = 1$. This set represents in the transmission graph an induced matching.
- **(f)** Thus the last time slot does not influence in this example the choice of the active set.




13 12 14 10 13 12 8
13 13 2
9 4 11 4 6
3 12 15 7 9 4 10 4 15 10
10 9 14
13 10 9
13 12 1 9 14 9 5
9 8 1
12
11 7 5 12 13 11 6 13 12 14
8 14 9 15 10


13 12 14 10 13 12
13 13 2 8
9 4 11 4 6
15
3 12 7 9 4 10 4 10
10 15
9 14
13 10 9
13 12 1 9 5
14
9 8 1
12
11 7 5 12 13 11 13 12
6 14
8 14 9 15 10

13 12 14 10 13 12
13 13 2 8
9 4 11 4 6
15
3 12 7 9 4 10 4 10
10 15
9 14
13 10 9
13 12 1 9 5
14
9 8 1
12
11 7 5 12 13 11 13 12
6 14
8 14 9 15 10

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399