Contention Management for 5G

DB,CC,MG,OM,YS

February 8, 2017

Abstract

This article treats about Contention Management for 5G.

1 Introduction

- Context and problematic
- Related works
- Article contribution

2 Model

2.1 Definitions

We consider a directed graph $G = (V, A)$ modelling a network. Each arc (u, v) in A is labeled by an integer $Dl(u, v) \geq 1$ that we call the delay and which represents the number of time slots taken by a signal to go from u to v using this arc. Note that for any arc (u, v), $Dl(u, v) = Dl(v, u)$.

A **route** r in G is a sequence of consecutive arcs $a_0, \ldots, a_{k-1}$, with $a_i = (u_i, u_{i+1}) \in A$.

The **latency** of a vertex u_i in r, with $i \geq 1$, is defined by

$$\lambda(u_i, r) = \sum_{0 \leq j < i} Dl(a_j)$$

We also define $\lambda(u_0, r) = 0$. The latency of the route r is defined by $\lambda(r) = \lambda(u_k, r)$.

A **routing function** $\mathcal{R}$ in G associates to each pair of vertices (u, v) a route from u to v. Let $\mathcal{C}$ be an **assignment** in G, i.e., a set of couples of different vertices of G. We denote by $\mathcal{R}_{\mathcal{C}}$ the set of routes $\mathcal{R}(u, v)$ for any (u, v) in $\mathcal{C}$. $\mathcal{R}_{\mathcal{C}}$ is called an **assigned graph**, and contains all the information needed in forthcoming problems (couples of vertices, routes and delay of the arcs).

TODO: Si on s'en sert, ajouter ici que le routage est cohérent.

2.2 Slotted time Model

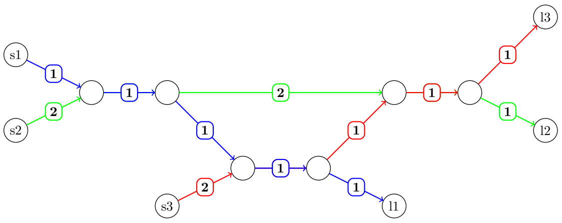
Consider now a positive integer P called the **period**. In our problem the messages we send in the network will be periodic of period P and thus we consider slices of time of P discrete slots. Assume we send a message at the source of the route r, at the time slot m in the first period, then a message will be sent at time m at each new period. We define the first time slot at which a message reaches a vertex v in this route by $t(v,r) = m + \lambda(v,r) \mod P$.

A message usually cannot be transported in a single time slot. We denote by τ the number of consecutive slots necessary to transmit a message. Let us call $[t(v,r)]$ the set of time slots used by r at a vertex v in a period P, that is $[t(v,r)] = \{t(v,r) + i \mod P \mid 0 \leq i \leq \tau\}$.

A P**-periodic affectation** of $\mathcal{R}_{\mathcal{C}}$ consists in a set $\mathcal{M} = (m_0, \dots, m_{c-1})$ of c integers that we call **offsets**, with c the cardinal of the assignment $\mathcal{C}$. The number m_i represents the index of the first slot used in a period by the route $r_i \in \mathcal{R}_{\mathcal{C}}$ at its source. A P-periodic affectation must have no **collision** between two routes in $\mathcal{R}_{\mathcal{C}}$, that is $\forall (r_i, r_j) \in \mathcal{R}_{\mathcal{C}}^2, i \neq j$, we have

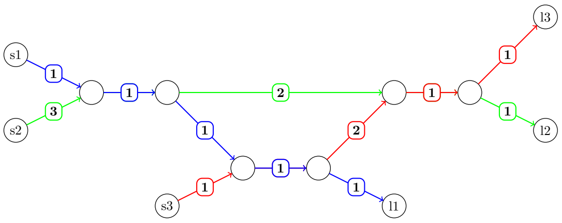
$$[t(u,r_i)] \cap [t(u,r_j)] = \emptyset.$$

Notice that the notion of P-periodic affectation **is not monotone** with regard to P. Indeed, consider messages of size 1, we can build a graph, with c routes $r_i \in \mathcal{R}_{\mathcal{C}}, i < c$ which all intersect two by two and such that if r_i and r_j have v as first common vertex we have $\lambda(v,r_i) - \lambda(v,r_j) = 1$. Therefore there is a 2-periodic affectation by setting all m_i to 0.



$$\lambda(v,r_i) - \lambda(v,r_j) = 1$$

On the other hand if we set all $\lambda(v,r_i) - \lambda(v,r_j) = P$, there is no P-periodic affectation if $P < c$.



$$\lambda(v, r_i) - \lambda(v, r_j) = 2$$

Here for P=2, there is no P-periodic affectation.

Therefore if we choose and odd value for P and $c = P + 1$, if there is no P-periodic affectation but modulo 2 all $\lambda(v, r_i) - \lambda(v, r_j)$ are equal to one thus we have a 2-periodic affectation.

2.3 Problems

The main theoretical problem we have to deal with in this context is the following.

Problem Periodic Routes Assignment (PRA)

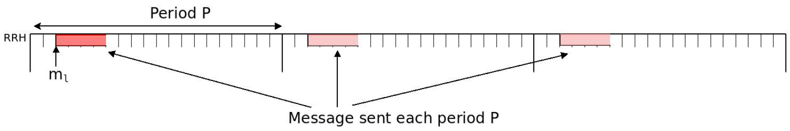
Input: a graph $G = (V, A)$, a set $\mathcal{C}$ of pair of vertices, a routing function $\mathcal{R}$ and an integer P.

Question: does there exist a P-periodic affectation of $\mathcal{C}$ in $(G, \mathcal{R})$?

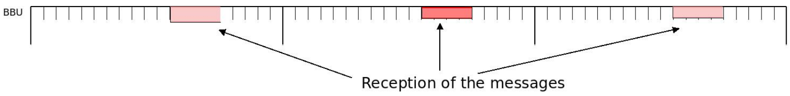
We will prove in Sec. 3 that the Problem PRA is NP-complete, even in restricted settings. In fact, even approximating the smallest value of P for which there is a P-periodic assignment is hard.

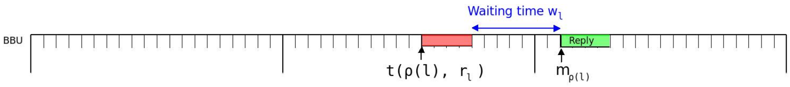
In the context of cloud-RAN applications, we consider here the digraph $G = (V, A)$ modeling the target network and two disjoint subsets of vertices S and L, where S is the set of BBU and L is the set of RRH. We denote by n the size of S and L. We are given a period P, a routing function $\mathcal{R}$ and a bijection $\rho : L \to S$ which assigns a BBU to each RRH. Let $\mathcal{C}_\rho = \{(l, \rho(l))\}_{l \in L} \cup \{(s, \rho^{-1}(s))\}_{s \in S}$. Let consider a P-periodic affectation of $\mathcal{C}_\rho$ which associates m_l to $(l, \rho(l))$ and $m_{\rho(l)}$ to $(\rho(l), l)$.

This affectation represents the following process: first a message is sent in l, through the route r_l, at time m_l.



This message is received by $\rho(l)$ at time $t(\rho(l), r_l)$. It is then sent back to l in the same period at time $m_{\rho(l)}$ if $m_{\rho(l)} > t(\rho(l), r_l)$, otherwise at time $m_{\rho(l)}$ in the next period. The time between the arrival of the message and the time it is sent back is called the **waiting time** and is defined by $w_l = m_{\rho(l)} - t(\rho(l), r_l)$ if $m_{\rho(l)} > t(\rho(l), r_l)$ and $w_l = m_{\rho(l)} + P - t(\rho(l), r_l)$ otherwise.





When a BBU receives a message, it must compute the answer before sending it back to the RRH. This time can be encoded in the last arc leading to the BBU and thus we need not to consider it explicitely in our model.

Thus, the whole proccess time for a vertex l is equal to

$$PT(l) = \lambda(r_l) + w_l + \lambda(r_{\rho(l)}).$$

The **maximum process time** of the P-periodic affectation $\mathcal{M}$ is defined by $MT(\mathcal{M}) = \max_{l \in L} PT(l)$. The problem we want to solve is the following.

Problem Periodic Assignment for Low Latency(PALL)

Input: a digraph G, a matching ρ from L to S two disjoint set of vertices of G, a routing function $\mathcal{R}$, a period P, an integer T_{max}.

Question: does there exist a P-periodic affectation $\mathcal{M}$ of $\mathcal{C}_\rho$ in $(G, \mathcal{R})$ such that $MT(\mathcal{M}) \leq T_{max}$?

3 Solving PRA

3.1 NP-Hardness

In this section we assume that the size of a message τ is equal to one, which implies hardness of PRA and PALL for all τ. Consider an instance of Problem PRA, i.e., a digraph $G = (V, A)$, an assignment $\mathcal{C}$, a routing function $\mathcal{R}$ and a period P. The **conflict depth** of a route is the number of other routes which share an edge with it. The conflict depth of an assignment $\mathcal{C}$ is the maximum of the conflict depth of the routes in $\mathcal{R}_{\mathcal{C}}$. The **load** of an assignment is the maximal number of routes sharing the same arc. It is clear that a P-periodic affectation must satisfy that P is larger or equal to the load.

We give two alternate proofs that PRA is NP-complete. The first one works for conflict depth 2 and is minimal in this regards since we later prove that for conflict depth one, it is easy to solve PRA. The second one reduces the problem to graph coloring and implies inapproximability when one tries to minimize the parameter P.

Proposition 1. *Problem PRA is* NP*-complete, when the routing is of conflict depth two.*

Proof. The problem PRA is in NP since given an offfset for each route in an affectation, it is easy to check in linear time in the number of edges whether there are collisions.

Let $H = (V, E)$ be a graph and let d be its maximal degree. We consider the problem to determine whether H is edge-colorable with d or $d + 1$ colors. The edge coloring problem is NP-hard [?] and we reduce it to PRA to prove its NP-hardness. We define from H an instance of PRA as follows. The graph G has for vertices $V' = \{v_1, v_2 \mid v \in V\} \cup \{l_{u,v}, s_{u,v} \mid (u, v) \in E\}$

that is two vertices for each vertex and for each edge of H. Let A bet the set of arcs of G, defined by

$$A = \{(v_1, v_2) \mid v \in V\} \cup \{(u_2, v_1) \mid u \neq v \in V^2\} \cup \{(l_{u,v}, u_1), (v_2, s_{u,v}) \mid (u, v) \in E\}.$$

All these arcs are of weigth 0. For each edge $(u, v) \in E$, the is a route $r_{u,v} = s_{u,v}, u_1, u_2, v_1, v_2, l_{u,v}$ in $\mathcal{R}$. The affectation $\mathcal{C}$ is the set of pair of vertices $(s_{u,v}, l_{u,v})$.

Observe that the existence of a d-coloring of H is equivalent to the existence of a d-periodic affectation for $(G, \mathcal{R}, \mathcal{C})$. Indeed, a d-coloring of H can be seen as a labeling of its edges by the integers in $\{0, \ldots, d-1\}$ and we have a correspondance between a d-coloring of H and offests for the routes of $(G, \mathcal{R}, \mathcal{C})$. By contruction, the constraint of having no collision between the routes is equivalent to the fact that no two adjacent edges have the same color. Therefore we have reduced edge coloring to PRA which concludes the proof. □

TODO: Faire un dessin d'illustration ?

Remark that we have used zero weigth in the proof. If the weigths must be strictly positive, which makes more sense in our model since we model the latency of physical links, it is easy to adapt the proof. We just have to set them so that in any route the delay at u_1 is equal to d and thus equal to 0 modulo d. We now lift this hardness result to the problem PALL.

Corollary 1. *Problem PALL is* NP*-complete for routing of conflict depth two.*

Proof. We consider $(G, \mathcal{R}, \mathcal{C}, P)$ an instance of PRA such that no element appears both in the first and second position in a pair of $\mathcal{C}$. Remark that this condition is satisfied in the previous proof, which makes the problem PRA restricted to these instances NP-complete. Let us define $T_{max} = 2 \times \max_{r \in \mathcal{R}} \lambda(r) + P$. We define ρ as the function which maps u to v when $(u, v) \in \mathcal{C}$. The instance $(G, \mathcal{R}, \rho, P, T_{max})$ is in PALL if and only if $(G, \mathcal{R}, \mathcal{C}, P)$ is in PRA. Indeed the waiting time of each route is by definition less than P and thus the maximal process time less than T_{max}. Therefore the fact that $(G, \mathcal{R}, \rho, P, T_{max})$ is in PALL is equivalent to the existence of a P-periodic assignment of $\mathcal{C}_\rho$ which is equal to $\mathcal{C}$. □

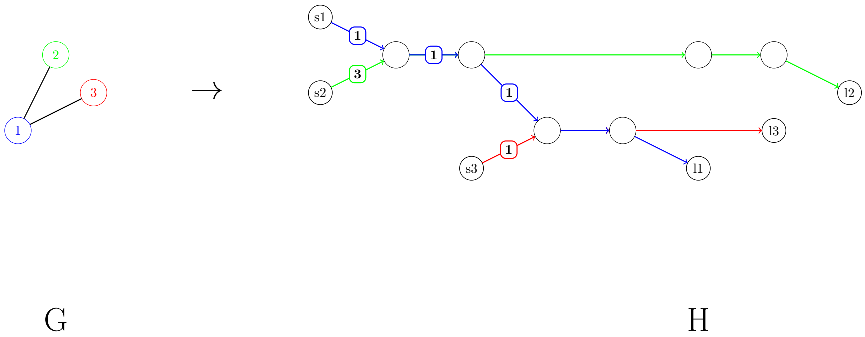
Let MIN-PRA be the problem, given a graph, a routing and an affectation, to find the minimal period P such that there is a P-periodic affectation.

Theorem 1. *The problem MIN-PRA cannot be approximated in polynomial time within a factor $n^{1-o(1)}$, with n the number of routes, unless* $\mathsf{P} = \mathsf{NP}$ *even when the load is two.*

Proof. We reduce graph coloring to PRA. Let H be a graph instance of the k-coloring problem. We define G in the following way: for each vertex v in H, there is a route r_v in G. Two routes r_v and r_u share an arc if and only if (u, v) is an edge in H; this arc is the only one shared by these two routes. All arcs are of delay 0.

Observe that the existence of a k-coloring of H is equivalent to the existence of a k-periodic affectation in G, by converting an offset of a route into a color of a vertex and reciprocally. Therefore if we can approximate the minimum value of P within a factor f, we could approximate the minimal number of colors needed to color a graph within a factor f, by doing the previous reduction for all possible k. The proof follows from the hardness of approximability of finding a minimal coloring [?]. □

In particular, this reduction shows that even with small maximal load, the minimal period can be large.



3.2 MIN-PRA

Exemple de cas polynomiaux

4 The Star Topolgy

In this section, we consider a particular case of the model, in which for each (u, v) , the route is the same in both directions. This means that $\mathcal{R}(u, v)$ uses the same arcs as $\mathcal{R}(v, u)$ in the opposite orientation.

4.1 Intro

PALL NP-Hard car PRA NP-Hard
Résultats valables sur Topologie 1 avec nos paramètres TODO: J'ai viré star affectation, car je pense qu'il n'y a rien à dire là dessus.

4.2 No waiting times

4.2.1 Shortest-longest

Algo

Period

4.2.2 Greedy Algorithm with higher bound

Algorithm 1 Greedy with Higher bound Period(GHP)

```
Input: $\mathcal{R}_{\mathcal{C}}$, period $P$
Output: A P-periodic affectation in p $\leq P$, or FAILURE
  $P1[P]$ slots of size $\tau$ in first way period.
  $P2[P]$ slots of size $\tau$ in back way period.
  for all route i in $\mathcal{R}_{\mathcal{C}}$ do
    for all slot j in P1 do
      if $P1[j]$ is free AND $P2[j + \lambda(r_i)]$ is free then
        $o_i \leftarrow j$
      end if
      if No intervals are found for i then
        return FAILURE
      end if
    end for
  end for
```

Period This algorithm gives us a solution without waiting times in a maximum period $3.\tau.c$, if we have c routes.
Suppose that we have a period of $3.\tau.c$ slots, divided in τ macro-slots. Let us call *forward period* the period in the central node when the messages goes to RRH to BBU, and *backward period* the period when the messages comes back in the other way. Put a message in the first slot of size τ in the forward period, such that the corresponding area in the backward period is free.This message takes at most 2 slots of time τ in the backward period.
TODO: Dessin qui illustre ça
When $k < c$ messages are put in the forward period, and we want to add another message, there is $3.c - k$ free slots of size τ in the forward period. Those $3.c - k$ gives us $3.c - k$ possible slots in the backward periods.
The k messages uses at most $2k$ slots of size τ used in the backward period. Since $k < c$, $2k < 3.c - k$, thus using the pigeonhole principle, there is at least one free slot in the backward period for the new message.

4.2.3 Exhaustive generation

Décrire l'algo, expliquer les coupes

4.2.4 Results

Resultats des simulations : Shortest-longest optimal pour ces parametres.

4.3 Allowing waiting times

4.3.1 Intro

Importance des waiting times quand la période est donnée (Résultats D'éxepriences et preuve avec l'exemple)

4.3.2 LSG

Algorithm

Analysis Parler de LSO et expliquer pourquoi LSG mieux avec nos params

4.3.3 Results

Random

Distributions

5 Conclusion

References