Contention Management for 5G

DB,CC,MG,OM,YS

January 31, 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 symmetric 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 $\mathcal{C}$. 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 $\mathcal{M} = (m_0, \ldots, m_{c-1})$ 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

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

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 at time $\bar{m}_l$, which is in the same period if $m_{\rho(l)} > t(\rho(l), r_l)$ otherwise it is in the next. 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 BBU. This time is taken into account 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

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 an assignment $\mathcal{C}$ is the number of arcs crossed by at least two routes in $\mathcal{R}_\mathcal{C}$; in other words it is the number of potential conflicts between these routes. The *load* of an assignment is the maximal number of routes sharing a same arc (i.e., the congestion of this 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.

Proposition 1. *Problem PRA is* NP*-complete, for a routing with conflict depth two.*

Proof. Let $H = (V, E)$ be a graph and d its maximal degree. We want to determine whether H is edge-colorable with d or $d + 1$ colors. We reduce this edge coloring problem to PRA. We define from H an instance $G, \mathcal{C}, \mathcal{R}, P$ as follows. For each $v \in V$ there are two vertices in G connected by an arc (v_1, v_2) and none of these arcs are incident. The delay of these arcs is equal to 1. The set $\mathcal{C}$ is made of couples of vertices $< s_{u,v}, l_{u,v} >$ for each edge $[u, v]$ in H and the routes associated is $\mathcal{R} = s_{u,v}, u_1, u_2, v_1, v_2, l_{u,v}$. The delays of arcs $(s_{u,v}, u_1)$ and $(v_2, l_{u,v})$ are equal to $d(d + 1) - 1$.

Let ϕ be an edge coloring with k colors of H. We can build a k-periodic affectation of $\mathcal{C}$ in $(G, \mathcal{R})$ by assigning to the source $s_{u,v}$ of each route an offset of $\phi(s_{u,v})$. Indeed, if two routes r_1 and r_2 in $\mathcal{R}_{\mathcal{C}}$ share the same edge, say (v_1, v_2) then they represent two edges e_1 and e_2 of H incident to the vertex v. Therefore $\lambda(v_1, r_1) = \phi(e_1) \mod k$ because the delays of the edges before v_1 sum to $d(d+1)$ or $2(d(d+1))$ which are equal to 0 modulo k since $k = d$ or $k = d+1$. Thus $\lambda(v_1, r_1) - \lambda(v_1, r_2) \mod k = \phi(e_1) - \phi(e_2) \mod k$ and $\lambda(v_1, r_1) - \lambda(v_1, r_2) \mod k > 0$ since $\phi(e_1) \neq \phi(e_2)$.

Now consider a k-periodic affectation of $\mathcal{C}$ in $(G, \mathcal{R})$. For each $[u, v]$ in H, we define $\phi(u, v)$ to be the offset of the route beginning at $s_{u,v}$ in $\mathcal{R}_{\mathcal{C}}$. For the same reasons as in the last paragraph, ϕ is an edge coloring with k colors. Therefore we have reduced edge coloring which is NP-hard [**?**] to PRA which concludes the proof. □

Solving Problem PALL implies to solve PRA for an assignment $\mathcal{C}_\rho$ defined on S and L (see Section 2). Thus, we have the following corollary :

Corollary 1. *Problem PALL is NP-Hard.*

As another corollary of the previous proposition, given G, $\mathcal{R}$, P and two disjoint subsets S and L in G, the problem of knowing if there is a matching ρ from L in S such that the answer to problem PRA for instance $G, \mathcal{C}_\rho, \mathcal{R}, P$ is NP-hard since checking a potential feasible solution for this problem implies to solve PRA.

Consider now MIN-PRA as the minimisation problem related to PRA, consisting in minimizing the peril P.

Theorem 1. *Problem MIN-PRA cannot be approximate within a factor $n^{1-o(1)}$ unless $\mathsf{P} = \mathsf{NP}$ even when the load is two and n is the number of vertices.*

Proof. We reduce PRA to graph coloring. 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. We put delays inbetween shared arcs in a route so that there is a delay k between two such arcs.

As in the previous proof, a k-coloring of H gives a k-periodic affectation in G and conversely. 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 fator 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 simple

4 Proposed Solutions, solving PALL

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 Exhaustive generation

Décrire l'algo, expliquer les coupes

4.2.3 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