Latency for periodic multicasting

DB, CC, OM, YS, MG

May 18, 2016

1 Introduction

1.1 Context

Mobile networks are controled by many radio stations, composed of 2 major parts :

1. Base Band Units (BBU) : are responsible for computing the signals, and communicate with the core network.
2. Remote Radio Head (RRH) : are the antennas and have only to communicate with the mobile devices.

The part of the network between BBU and the core network is called backhaul.

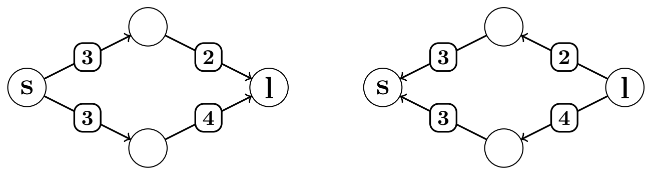
The operators are searching for a way to dissociate BBU from RRH, and regroup many BBU in some data centers, that would allow easier updates and maintenance. Nevertheless such a splitting enforces hard throughput and latency constraints on the communication network between BBU and RRH. This part of the network is called fronthaul.

In this article, we will work on latency constraints in fronthaul.

1.2 Definitions

We consider a directed graph $G = (V, A)$ with two non intersecting subsets of vertices: a subset L of nodes which are called *leaves* and a subset S of nodes which are called *sources*. The indegree of the node in S is equal to 0, and the outdegree of the node in L is also equal to 0. We denote by $\mathcal{L}$ the cardinal of L. The digraph G models the network, S the possible BUU and L the set of RRH. Each arc (u, v) in A is has an integer weight $Dl(u, v) \geq 1$ representing the time taken by the signal to go from u to v by using this arc.

We consider $G = (V, A)$ and $G^r = (V, A^r)$ wherein the set of vertices is the same and A^r represents the edges of A directed in the other way.



$G = (V, A)$ $G^r = (V, A^r)$

Note that there is a special case in which the graph may have oriented cycles (optical ring topologie), otherwise, there is no oriented cycles.

A route r is a sequence of arcs $a_0, \ldots, a_{n-1}$, with $a_i = (u_i, u_{i+1}) \in A$ such that $u_0 \in S$ and $u_n \in L$. The *latency* of a vertex u_i in r, with $i \geq 1$, is defined by

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

We also define $\lambda(u_0, r) = 0$. The latency of the route r is defined by $\lambda(r) = \lambda(u_n, r)$. In graph theory, a route is a simple path in the graph, and its latency is its weight.

A **routing function** $\mathcal{R}$ is an application associating a route $\mathcal{R}(s, l)$ to each couple $(s, l) \in S \times L$ in G. Moreover $\mathcal{R}$ satisfies the *coherent routing* property: the intersection of two routes must be a path.

For simplicity, we assume that we have as many source nodes as we have leaves ($\mathcal{S} = \mathcal{L}$). A **$\mathcal{R}$-matching** is a bijection $\rho : S \to L$ which associates to each $s_i \in \{s_0, ..., s_{\mathcal{S}-1}\}$ a $l_i \in \{l_0, ..., l_{\mathcal{L}-1}\}$ using a route in $\{r_0, \ldots, r_{\mathcal{L}-1}\}$. This means that each route in ρ is defined by its source node and its leave node.

For each arc $(u, v) \in A$, we denote by $\rho(u, v)$ the subset of routes of ρ containing (u, v). We define the load of an arc (u, v) as the number of routes that use this arc thus : $load(u, v) = |\rho(u, v)|$.

The quintuplet $N = (G, S, L, R, \rho)$ defines a **matched graph**. We call N^r the quintuplet (G^r, S, L, R, ρ^r), where ρ^r is the $\mathcal{R}$-matching obtained using the same routes, with inverted arcs.

Let $P > 0$ be an integer called *period*. A P-periodic affectation of N, a matched graph consists in a set $\mathcal{M} = (m_0, \ldots, m_{\mathcal{L}-1})$ of $\mathcal{L}$ integers that we call *offset*. Each time window is divided in P slots and the number m_i represents the slot number used by the route r_i at its source. We define the time slot used by a route r_i at any vertex v of the route by

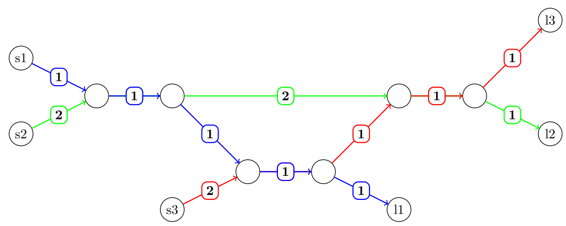
$$t(v, r_i) = m_i + \lambda(u, r_i)) \mod P.$$

A P-periodic affectation must have no *collision* between two routes in ρ, that is $\forall r_i, r_j \in \rho, i \neq j$, two routes intersecting in u, and containing the same arc (u, v), we have

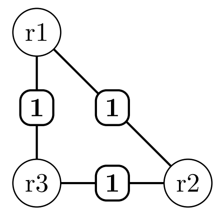
$$t(u, r_i) \neq t(u, r_j).$$

We define $\alpha(u, r_i, r_j) = |\lambda(u, r_i) - \lambda(u, r_j)|$. This will be useful later, when we'll introduce route conflict graphs.

Notice that the notion of P-periodic affectation **is not monotone** with regard to P. Indeed, we can build a $\mathcal{R}$**-matching** of a graph, with l routes $r_1, \ldots, r_l$ 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.

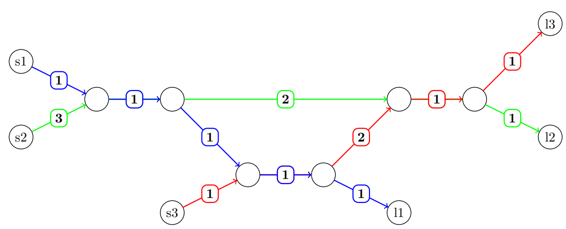


$\lambda(v, r_i) - \lambda(v, r_j) = 1$

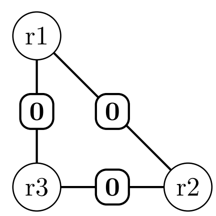


Conflict graph

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



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



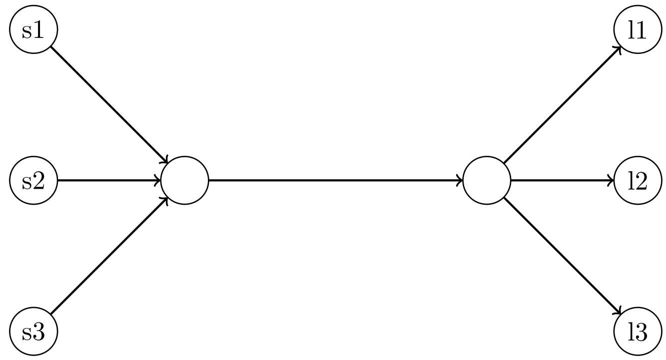
Conflict graph

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

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

1.3 Real Network Topologies

To illustrate we consider a basic network topology, composed of some base stations, represented by source nodes S, all connected to the same switch, which will be a vertex, connected himself to another vertex, corresponding to a switch, connected to some leave nodes L representing the Antennas.



Example of topologie 1.

TODO: ajouter les autres topologies

1.4 Fronthaul networks

The application we address here by studying the problems defined above is the following. Consider a fronthaul network in which source nodes in S represent computing units in data centers, each one able to do a remote process for base stations modeled by leaves node in L. Consider a $\mathcal{R}$-matching ρ of S in L. Consider a leave l, its dedicated source node s and $R(l,s)$ the route from l to s in R. We consider a P-periodic affectation of N, and also another P-periodic affectation of N^r. The periodic process is then the following one:

1. During each period of duration P, leaves l send a message to their dedicated sources s, using their dedicated routes and considering the P-periodic affectation of N.

2. When s receives this message, it makes a computation to answer l. this computation time is set.

3. Then sources s sends a message to their leaves l considering the P-periodic affectation of N^r.

On source nodes, between the end of computation time and the emission time of the message (given by the P-periodic affectation), there is a waiting time. We define by $\omega : r \rightarrow \mathbb{N}$ the waiting time of a route r in the $\mathcal{R}$-matching considered, i.e. the time during while the message is "sleeping", waiting to be sent through the network.

Source and leave nodes periods have to be the same. Indeed, leave nodes have to receive a message at every period, so source nodes have to send a message at every same period. So, we will find two P periodic affectations from the same P in both ways. If the message have to be buffered, we want to do it in souces nodes. We define by θ the computation time required at the source node before sending an answer to it's leave node.

Let us call $T(r)$ the process time on a route r:

$$T(r) = 2\lambda(r) + \omega(r) + \theta(r)$$

We consider 3 main problems:

Problem Periodic Routes Assignment (PRA)
Input: matched graph N, integer P.
Question: does there exist a P-periodic affectation of N ?
Optimisation goal: minimizing P.

This problem is our basic problem. We need to find the offset for each routes such that there is no collision between the signals emitted by sources at any nodes in the graph.

Problem Network Periodic Assignment (NPA)
Input: $G = (V, A)$, S, L, a routing $\mathcal{R}$, integers P
Question: does there exist a $\mathcal{R}$-matching ρ of S in L such that there exists a P-periodic affectation $\mathcal{M}$ of ρ ?
Optimisation goal: minimizing P.

This problem is the same problem than PRA in which is added the liberty to find the $R-matching$.

In our network application θ is set to 2,6ms, the period P is 1ms and T(r) must be less or equal than 3ms.

So P and the $R-matching$ are given, we do not need to minimize P and P is generally great enough to carry the dataload. Therefore we want to optimize the time taken by the message to do the two way trip.

A **2-way-trip affectation** $X = (x_0, ..., x_{\mathcal{L}-1})$ of N is a P-periodic affectation $\mathcal{M} = (m_0, ..., m_{\mathcal{L}-1})$ of N^r, then a second P-periodic affectation $\Omega = (\omega_0, ..., \omega_{\mathcal{L}-1})$ of N considering the computation time θ, such that, $\forall i \in [0, ..., \mathcal{L} - 1]$:

$$x_i = (m_i + \lambda(r_i) + \theta + \omega_i) mod P.$$

Note that i designe the route and that the second P-periodic affectation Ω correspond to the waiting times of routes : $\omega_i = \omega(r_i)$.

The real network problem is the following:

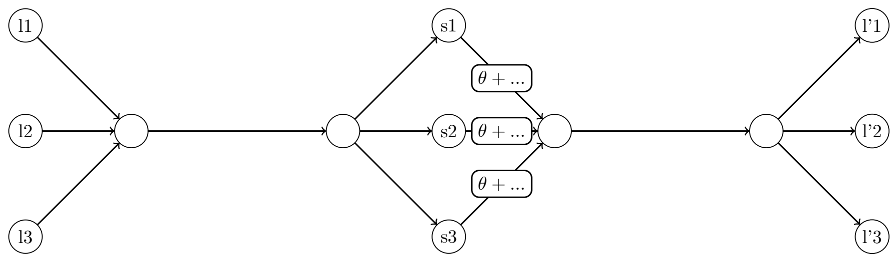
Problem Periodic Assignment for Low Latency(PALL)
Input: matched graphs N, integer P, T_{max}, θ.
Question: does there exist a 2-way-trip affectation of N, such that $\forall r \in \rho$, $T(r) \leq T_{max}$.
Optimization goal 1: minimizing $\sum_{r \in \rho} T(r)$.

By minimizing the sum of all the routes, we allow a better global Quality of Service trough the network.

Optimization goal 2: minimizing max(T(r)).

2 Waiting time usefulness

Here, we study the impact of the waiting time parameter. In à first time, we could consider our model like a more complex graph in wich the two way trip process is a single trip of this new graph.



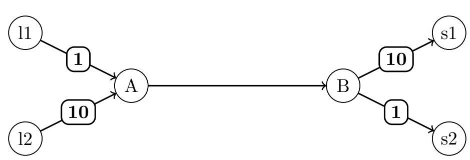
This model involve to set all the waiting times to 0, and to find only one P-periodic affectation, that choose the first offsets which represents the emission time of each routes. So, if all the waiting times are equal to 0, and we can find this P-periodic affectation, this means that only the delay $\Lambda(r)$ of each routes and the computation times θ will cost in our travel times. Since we can not minimize it (unless we use more powerful equipment), our problem would be sovled, and equivalent to PRA.

Note that such a model does not respect the coehrent routing property. Therefore, we can not consider this model but the question is still : can we find a solution without waiting times.

Lemma 1. *Solving PRA in such a graph significatively increase the period P.*

Proof. The folowing example shows us that you can find a P-periodic affectation such that there is no waiting time, but it needs a greater time window P than a solution with waiting times.

Consider the graph :



The middle link can be represented without weight, because it is common to all the routes, so it can be simplified in calculations. Likewise, we do not consider θ, the calculation time.

We consider messages of 10 slots. If you search for a P-periodic affectation, wihtout using waiting times, u have 2 choices :

- l1 sends before l2 : the message from l2 will leave A after 11 slots. Let us call X the minimal time slot in wich l2 can emmit. To avoid colisions with the message from l1 in a , $X \geq 1$. In the other way, the message from l1 will use B from time slot 21 to 21+10. So, even if $X = 1$, message from l2 uses B in the sames time slots than message from l1 (the message crosses B during time slots 12 to 12+10). So $X + 12 \geq 21 + 10 \rightarrow x \geq 19$. Then if we take $X = 19$, our time windows is at least 19 slots.

- l2 sends before l1: at the first node A, l2 will have to wait 10+10 time slots to send his message, so the windows will be at lest 20.

If we allow waiting times: there is an easy scheduling using only a 3 time slots windows : Sending message 1 at time 0, message 2 at time 1, so the messages will not cross A in the same time. to be sent back, message 2 comes back immediatly and is using B from slots 13 to 23,then message 1 is using B from slots 23 to 33, waiting 2 time slots at s1.

So time windows is $[0, 1, 2]$ widely smaller than best solution without waiting times.

□

3 Complexity results

The *conflict depth* of a $\mathcal{R}$-matching is the maximum number of arcs in a route which are shared with other routes. In other words it is the maximal number of potential conflicts along one route. The *load* of a $\mathcal{R}$-matching is the maximal number of routes which share the same edge. It is clear that a P-peridodic 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 minimimal 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. In all this section the reservation δ **is assumed to be** 0.

TODO: ajouter des références à des problèmes de réseau similaires et de transport en commun qui sont également NP complets.

We define two characteristic graphs associated to a $\mathcal{R}$-matching.

Definition 1 (Edge Conflict Graph)**.** *The Edge Conflict Graph (ECG) of a $\mathcal{R}$-matching of a digraph $G = (V, A)$ is the graph $G' = (V', E')$ such that a vertex of V' is an arc of A with a load greater than one. There is an edge between two vertices of V' if and only if there is a route which contains the two arcs of A corresponding to the two vertices.*

This graph can be oriented so that an edge is oriented as the path in a route it represents and we can associate as weight to this edge, the delay of the corresponding path.

Definition 2 (Route Conflict Graph)**.** *The Route Conflict Graph (RCG) of a $\mathcal{R}$-matching of a graph $G = (V, E)$ is the graph $G' = (V', E')$ where the vertices of V' are the routes of the $\mathcal{R}$-matching and there is an edges between two vertices corresponding to two routes if and only if they share an edge.*

We fix an arbitrary ordering on the elements of $V' = \{v_1, \ldots, v_n\}$. We can associate a weight to each edge (v_i, v_j). Assume that $i < j$, and that the routes r_i and r_j corresponding to v_i and v_j have u as first common vertex. Then the weight of (v_i, v_j) is $\lambda(u, r_i) - \lambda(u, r_j)$.

TODO: Dire comment une solution à PRA s'exprime dans ces graphes avant de faire les preuves de NP-complétude

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

Proof. Let $G = (V, E)$ be a graph and d its maximal degree, we want to determine whether it is edge colorable with d or $d + 1$ colors. We reduce this edge coloring problem to PRA: we define a graph H, for each $v \in V$ there are two vertices connected by an edge (v_1, v_2) and none of these edges are incident. Those edges are of weight 1. The schedule ρ of H is the set of routes $s_{u,v}, u_1, u_2, v_1, v_2, l_{u,v}$ for each edge (u, v) in G. The weight of the two new edges is $d(d + 1) - 1$. The parameter δ is set to 0.

Let ϕ be an edge coloring with k colors of G. We can build a k periodic schedule 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 share the same edge, say (v_1, v_2) then they represent two edges e_1 and e_2 of G 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 ρ. For each (u, v) in G, we define $\phi(u, v)$ to be the offset of the route beginning at $s_{u,v}$. 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 [2] to PRA which concludes the proof. □

Theorem 2. *Problem 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 G be a graph instance of the k-coloring problem. We define H in the following way: for each vertex v in G, there is a route r_v in H. Two routes r_v and r_u share an edge if and only if (u, v) is an edge in G and this edge is only in this two routes. We put weight inbetween shared edges in a route so that there is a delay k between two such edges.

As in the previous proof, a k-coloring of G gives a k-periodic schedule of H and conversly. Therefore if we can approximate the value of PRA 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 [4]. □

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

TODO: By using ideas similar to Vizing theorem, we may prove the following theorem for graph of congestion depth 2. In the proof there is just an analog of the lemma used to prove Vizing theorem, we should try to complete the proof. Also can we say something similar when the congestion depth is larger ?

Proposition 2. *The solution to PRA is either the load or the load plus one. Moreover, a solution of load plus one can be built in polynomial time.*

Proof. First we define an alerning path and how it characterizes an optimal solution of PRA. Let v be a vertex of degree d in the congestion graph (to be defined). We assume that the coloring of the edges is optimal (to define in the case of a congestion graph). The color α is not used in it neighborood. For every edge of color β from this vertex we build an $\alpha - \beta$ path, that is a maximal path $u_0, u_1, \dots, u_l$ so that (u_0, u_1) is of color $\beta, \beta + \lambda(u_0, u_1)$, then (u_1, u_2) is of color $\alpha + \lambda(u_0, u_1), \alpha + \lambda(u_0, u_2)$ and so on (changer λ car on ne suit pas des routes).

A maximal $\alpha - \beta$ path must be a cycle or the coloring is not minimal. □

As a corollary of the previous proposition, NPA is NP-hard since checking a potential feasible solution for NPA implies to solve PRA. Also, PALL is NP-Hard because solving it implies to slove PRA in both ways with a given P (1ms).

TODO: we may try to find where those problems are in the polynomial hierarchy, in the second level ?

4 Coloring and P-periodic affectation

To the route conflict graph, we can associate a system of inequations representing the constraints that the P-periodic affectation must satisfy.

Definition 3 (Conflict system)**.** *Let G be a weighted RCG, we associate to each vertex v_i of G the variable x_i. The conflict system is the set of inequations $x_i \neq x_j + w(e)$ for $i < j$ and $e = (v_i, v_j)$ edge of G.*

It is simple to see that the conflict system of a $\mathcal{R}$-matching has a solution modulo P if and only if the $\mathcal{R}$-matching has a P-periodic affectation. This kind of system can be solved by SAT solver or CSP solver and those two methods will be investigated on practical instances. TODO: Donner la formulation pour ces solvers et donner le résultat d'expérience dans une partie indépendante. Les comparer avec un solveur de notre cru avec quelques heuristiques simples.

Definition 4 (Additive coloring)**.** *Le G be a weighted graph, we say that G has an additive coloring with p colors if and only if its associated conflict system has a solution over $\mathbb{Z}/p\mathbb{Z}$. The minimal p for which the system has a solution is the additive chromatic number of G denoted by $\chi_+(G)$.*

Finding an optimal additive coloring is thus the same as solving our problem of finding a P-periodic affectation and is at least as hard as finding an optimal coloring. Because of the constraints on the edges, the additive chromatic number of a graph may be much larger than its chromatic number. When the graph is a clique both additive and regular chromatic number may be the size of the graph. However, we will now prove that for bipartite graphs the chromatic number is two while its additive chromatic number is arbitrary large.

We have the following values obtained by exhaustive search.

TODO: généraliser chi+ aux graphes sans poids en faisant un max ?

Fact 1. *The values we list here are the maximal ones we obtain by weighting bipartite graphs.*

1. $\chi_+(K_{2,2}) = \chi_+(K_{3,3}) = 3$ *with weight* $0, 1$
2. $\chi_+(K_{3,4}) = \chi_+(K_{3,5}) = 3$
3. $\chi_+(K_{3,6}) = 4$
4. $\chi_+(K_{4,4}) = \chi_+(K_{4,5}) = \chi_+(K_{4,6}) = 4$
5. $\chi_+(K_{5,5}) =?$

A nice theoretical question would be to find the way to put weights on a bipartite graph so that its additive chromatic number is maximal. My conjecture is that a $K_{l,l}$ can have an additive chromatic number in $O(l)$. The question is also interesting when we restrict the weights to $0, 1$.

Theorem 3. *There is a weighting of $K_{l^2,\binom{l}{l^2}}$ such that $\chi_+(K_{l^2,\binom{l}{l^2}}) > l$.*

Proof. Let V_1, V_2 be the bipartition of the graph we build and let $|V_1| = l^2$. For all $S \subseteq V_1$ with $|S| = l$, we denote by $v_1, \dots, v_l$ its elements, there is a single element v_S in V_2 which is connected to exactly the elements of S and such that the weight of v_S, v_i is $i - 1$. Because of this construction, V_2 is of size $\binom{l}{l^2}$. Moreover for any additive coloring of the graph we have constructed, a set of l elements in V_1 cannot have all the same color. But by the extended pigeon principle, since there are l^2 elements in V_1 at least l amongst them must have the same color. This prove that the graph we have built cannot have an additive coloring with l colors. □

The theorem can be improved so that the number of colors is logarithmic in the size of the bipartite graph.

Theorem 4. *There is a weighting of $K_{l^2,\binom{l}{l^2}}$ such that $\chi_+(K_{l^2,\binom{l}{l^2}}) > l$.*

Proof. Let $\mathcal{F}$ be a family of perfect hash functions from $[l^2]$ to $[l]$. It means that for any subset S of size l of $[l^2]$, there is an hash function $f \in \mathcal{F}$ such that f_S is injective. The construction of the bipartite graph is similar to the previous proof. Let V_1, V_2 be the bipartition of the graph we build and let $V_1 = \{v_1, \ldots, v_{l^2}\}$. For each function $f \in \mathcal{F}$ there is a vertex $v_f \in V_2$ and the weight of (v_i, v_f) is $f(i)$. By [3, 1] there is a family of perfect hash functions of size $2^{O(l)}$ therefore V_2 is of size $2^{O(l)}$. Again by using the pigeon principle and the perfect property of the family of functions, we prove that no additive coloring with l colors is possible. □

TODO: explain the removal of low degree vertices = kernelization + heuristic on the degree Also implement it in the solver

TODO: Comprendre la valeur sur d'autre familles de graphe, notemment celles qu'on peut rencontrer en pratique, par exemple petite tree width

5 Special cases study

After proving the complexity of problem PRA and NPA in the general case, we will now study special cases of problem PRA where its complexity is polynomial. First we define the notion of coherent routing.

As a consequence, for each node u in V, the subgraph of G induced by all the routes from u to all the other nodes in G is a tree. In the following, we only deal with coherent routing functions.

5.1 The disjoint paths PRA problem : DP-PRA

In this restriction of PRA, we are given a $\mathcal{R}$-matching with the following properties:

1. There are no common arcs for routes originating from different sources
2. The routing $\mathcal{R}$ is coherent

Proposition 3. *Problem DP-PRA can be solved in linear time according to the size of A.*

The first property of the DP-PRA problem ensures that an arc cannot belong to two routes in $\mathcal{R}$ originating from different sources in S. The second property ensures that if two routes originate from the same source x, they share the same arcs from x to a given vertex y and cannot share an arc after. This means that $\forall (u,v) \in A$, the arcs (u,v) with the highest load $l_{max} = max(load(u,v))$ are arcs a_0^j sharing a common origin $u_0 \in S$ and a P-periodic affectation for those arcs is a P-periodic affectation for all the arcs in A.

In a P-periodic affectation consists in a time schedule where routes on a same arc must be separated by a delay that is strictly superior to an integer $\delta \geq 0$. As a consequence, the minimum size of the period P is equal to $l_{max} \times (\delta + 1)$. This means that that on the arc with the highest load, we can schedule the first route at the moment $m_k = 0$, the second route at a moment $m_{k'} = \delta + 1$ and so on for all l_{max} routes.

5.2 The disjoint paths NPA-Problem : DP-NPA

In this subsection we study the NPA problem where in the graph induced by the the routing function $\mathcal{R}$:

- There are no common arcs for routes originating from different sources
- The routing $\mathcal{R}$ is coherent

Proposition 4. *Problem DP-NPA can be solved in linear time according to the size of A.*

The minimum size of P is obtained by minimizing the highest load of an arc a_0^j for any route $r^j \in \rho$. As all source vertices in S are connected to all sources vertices in L by a route in $\mathcal{R}$, a simple load balancing allows to obtain a maximum load equal to $max_{load} = \lceil \frac{\mathcal{L}}{|S|} \rceil$. Once the $\mathcal{R}$-matching is computed, we face the DP-PNA problem and the minimum value of P is thus equal to $max_{load} \times (\delta + 1)$.

5.3 The disjoint paths NPAC-Problem : DP-NPAC

In this problem, there is a delay constraint that must be satisfied by a $\mathcal{R}$-matching. This means that we must remove the routes in $r' \in \mathcal{R}$ where $\lambda(r') > K$.

Proposition 5. *Problem DP-NPAC can be solved in polynomial time according to the size of V.*

In a similar way to problem DP-PRA, the arcs with the highest load can only be the arcs a_0^j sharing a common origin $u_i \in S$. In order to minimize the size of the period P, we have to find a $\mathcal{R}$-matching such that the maximum number k_i of routes originating from a same source $u_i \in S$ is minimal. Having found this minimal value k, we face a problem equivalent to the DP-PRA problem.

In order to find a matching of vertices in S with vertices in L such that the maximum number of vertices in L assigned to a vertex $s_i \in S$ is inferior or equal to k, we will transform our problem in a flow problem.

5.3.1 Construction of a flow graph

Let us consider an instance $I = (G = (V, A), S, L, \mathcal{R}, \mathcal{P}, \delta, \mathcal{K}$ and $M)$ of NPA where the routes in $\mathcal{R}$ are only those that respect the delay constraint K. We first construct a complete bipartite graph $G' = (V', A')$ where V' is made of two sets of vertices : vertices V_1' corresponding to S and V_2' corresponding to L. A' is made of all possible arcs from vertices $v_1' \in V_1'$ to vertices $v_2' \in V_2'$, with a capacity 1. We then add to V' a source node S' and a sink node T'. Finally we includegraphics[scale=0.3]DP-NPA.pdf

Figure 1: Reduction of DPA-NPA into a flow problem

add to A' all the arcs from S' to each vertex $v'_1 \in V'_1$, with a capacity k and all the arcs from each vertex $v'_2 \in V'_2$ to T', with a capacity 1, where there is a route $\mathcal{R}(v'_1, v'_2)$. We have thus obtained a flow graph G' whose size is polynomial in regards to the size of G. We will now compute the maximum flow in G' in order to determine if its size is at least $\mathcal{L}$, in order to be able to connect all the leaves.

We can compute the size of a maximum flow in G' in a polynomial time using a generic flow algorithm as Ford-Fulkerson (it will terminate as arcs capacity are rational numbers). In order to minimize the objective k, we can begin with a value $k = 1$ and use a dichotomic approach to find the minimal value of k for which a maximal flow of size $\mathcal{L}$ exists in G'. The maximum value of k is M. If $k = M$ and the maximum flow value in $G' < \mathcal{L}$, then there is no valid $\mathcal{R}$-matching of S in L.

The complexity of minimizing k is thus $0(m^2 n \times log_2 n)$ where $m = |A|$ and $n = |V|$. We obtain in the end a $\mathcal{R}$-matching minimizing the maximal number of routes originating from a single source $u_s \in S$ and we are faced with the DP-PRA problem for this instance.

5.4 Intersecting paths problems for PRA

Next we study more general cases of PRA where routes originating from different sources can intersect. Let us consider an instance $I = (G, S, L, \mathcal{R}, \rho, \mathcal{P}, \delta)$ of PRA. We first define the collision induced graph of a $\mathcal{R}$-matching in $G = (V, A)$.

5.4.1 The 1-arc collision PRA problem

In this restriction of PRA, we are given a $\mathcal{R}$-matching with the following properties:

1. There is a bottleneck : a single arc $a_b = (u_b, u'_b)$ for which $load(a_b) \geq 1$ if the routes in $\rho(a_b)$ come from different sources in S
2. The routing $\mathcal{R}$ is coherent

Proposition 6. *Problem 1-arc collision PRA can be solved in linear time according to the size of A.*

In order to have a P-periodic affectation, the period P must be big enough for all arcs in $\rho(a_b)$, thus the minimal size of P is $P = load(a_b) \times (\delta + 1)$. Moreover, this is also the minimum solution for PRA. If we consider a scheduling $m_i \in \mathcal{M}'$ of each route $r_i \in \rho(a_b)$ such that there is a P-periodic affectation of this routes on the arc a_b with a size of $P = load(a_b) \times (\delta + 1)$, the schedule m_j in $\mathcal{M}$ corresponding to r_i will be $m_j = m_i - \lambda(u_b) mod P$. As the routing is coherent, no two routes originating from a same source can use different routes to attain

a_b thus the P-periodic affectation is valid for any arc $(u, v) \in A$ if it is valid on a_b.

5.5 The Data-center model

In this subsection we assume that we have a few datacenters which creates a few edges of high load in the routing graph. We further assume that to routes can share at most two edges with others routes, the first one with multiple routes at the exit of the datacenter and thes econd one with a single other route.

This model is a special case of conflict depth two and generalizes the disjoint path case. Its route conflict graph is very simple and better heuristics should work. Can we prove this case hard or easy ? TODO: Investigate this case and others arising from true data. For trees the problem is simple, what notion measure the proximity of a DAG to tree ? We could try the tree width ot the graph seen as not oriented, but it does not seem really adapted.

References

[1] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995.

[2] Ian Holyer. The np-completeness of edge-coloring. *SIAM Journal on computing*, 10(4):718–720, 1981.

[3] Jeanette P Schmidt and Alan Siegel. The spatial complexity of oblivious k-probe hash functions. *SIAM Journal on Computing*, 19(5):775–786, 1990.

[4] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 681–690. ACM, 2006.