Latency for periodic multicasting

DB, CC, OM, YS

April 22, 2016

1 Introduction

We consider a symmetric 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*. We denote by $\mathcal{L}$ the cardinal of L. The digraph G models the network, S the possible sources in the cloud and L the set of base-stations. Each arc (u, v) in A is caracterised by an integer delay $Dl(u, v) \geq 1$ representing the time taken by the signal to go from u to v by using this arc.

TODO: on devrait dire qu'un graphe dans le reste du texte est un digraphe symétrique donnée avec des sommets S,L et des poids sur les arêtes. Ça allégerait les phrases et les notations.

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. A **$\mathcal{R}$-schedule** of S in $L = \{l_0, \ldots, l_{\mathcal{L}-1}\}$ is a set of $\mathcal{L}$ routes $\rho = \{r_0, \ldots, r_{\mathcal{L}-1}\}$, in which for each $0 \leq j \leq \mathcal{L} - 1$, the last node of the route r_j is n_j. This means that each node $l_j \in L$ belongs to exactly one route in ρ. Moreover ρ satisfies the *coherent routing* property: the intersection of two routes must be a path. TODO: Je n'aime pas trop le nom R schedule qui est lourd et qui utilise le mot schedule alors qu'il n'y a pas encore de notion de temps mais juste de choix de station de départ. Il faudrait donner un nom à un graphe équippé d'une R schedule, ça serait pratique pour simplifier des choses dans la suite

The $\mathcal{R}$-schedule ρ defines a subgraph of G wich is the union of all paths in ρ that we call the *routing graph* and that we denote by G_ρ. Note that the routing graph may have oriented cycles.

TODO: Illustrer les r schedules par des dessins (un acyclique et un ring buffer). Est-ce que les cycles orientés arrivent vraiment en pratique et posent-ils des problèmes ?

For each arc $(u, v) \in A$, we denote by $\rho(u, v)$ the subset of routes of ρ containing (u, v).

Let $P > 0$ be an integer called *period*, and $\delta \geq 0$ be an integer called *reservation*. A P-periodic affectation of a $\mathcal{R}$-schedule ρ 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 for r_i and r_j in $\rho(u, v)$ assuming w.l.o.g. $t(u, r_i) < t(u, r_j)$, we have

$$\max(t(u, r_j) - t(u, r_i), t(u, r_i) - t(u, r_j) - P - 1) > \delta$$

TODO: On devrait peut-être plutôt définire $\lambda(u, r_i) - \lambda(u, r_j)$ comme un objet qui sera ensuite les poids du graphe des conflits

TODO: put a figure here to illustrate what it means to avoid a collision.

Notice that the notion of P-periodic affectation **is not monotone** with regard to P. Indeed, we can build a **$\mathcal{R}$-schedule** 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. On the other hand if we set all $\lambda(v, r_i) - \lambda(v, r_j) = P$, there is no P-peridodic affectation if $P < l$. 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.

TODO: Put here the figure corresponding to the remark + use this example in the definition as a running example for the conflict graphs and the conflict sytem

We consider two main problems:

Problem Periodic Routes Assignment (PRA)
Input: $G = (V, A)$, S, L, a routing $\mathcal{R}$ and a $\mathcal{R}$-schedule ρ of S in L, integers P and δ.
Question: does there exist a P-periodic affectation of ρ ?
Optimisation goal: minimizing P.

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

Later we might also consider a version of NPA with two additional constraints:

Problem Network Periodic Assignment with Constraints (NPAC)
Input: $G = (V, A)$, S, L, a routing $\mathcal{R}$, integers P, δ, K and M.
Question: does there exist a $\mathcal{R}$- schedule ρ of S in L such that

- there exists a P-periodic affectation $\mathcal{M}$ of ρ
- $\forall r_j \in \mathcal{R}$, $\lambda(r_j) \leq K$
- at most M routes in ρ can originate from the same source in S

Optimisation goal: minimizing P

TODO: Est-ce qu'un objectif raisonnable, plutôt que le problème précédent ne serait pas de minimiser la somme du plus long chemin plus P ? On peut imaginer des objectifs encore plus précis si on tiens compte du temps de traitement et de l'aller-retour. On pourrait aussi mettre des ressources sur chaque source et avoir un M différent pour chaque source.

TODO: Est-ce qu'on peut comparer ce problème au jeu spacechem ? peut-être queque'un a déjà fait une preuve sur ce jeu.

2 Reservation and PRA

Here we study the impact of the parameter δ (the reservation) in the PRA problem. Ideally we would like to get rid of it and find optimal solutions for any $\delta > O$ from a solution with $\delta = 0$ for the same graph. The most natural idea is to transform each slot in a time window into $(\delta + 1)$ slots to each and we multiply all offsets by $(\delta + 1)$ to get a periodic affectation of period multiplied by $(\delta + 1)$. Conversely, we could take any solution for $\delta > O$ and divide the offsets and the period by $\delta + 1$. Both ideas do not work because when we change the period the delays modulo the new period may change and not all in the same way.

TODO: Exemple ou ça marche bien: la clique

TODO: Donner des exemples de pourquoi ça ne marche pas

On the other hand, while modelling the problem, we arbitrarily chose what is of size one, thus we can always chose it so that it takes into account the reservation (we should validate this remark with Olivier). The problem is the fractional values we get for the delays (already for $\delta = 0$). By rounding them we could get transmission which seem at distance 1 in the time window but which are arbitrarily close. This could be a justification to always have $\delta = 1$.

TODO: On devrait mettre ça après le conflict system, car c'est plus simple à comprendre avec ce formalisme je pense.

3 Complexity results

The *conflict depth* of a $\mathcal{R}$-schedule 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 $\mathcal{R}$-schedule 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 $\mathcal{R}$-schedule.

Definition 1 (Edge Conflict Graph)**.** *The Edge Conflict Graph (ECG) of a $\mathcal{R}$-schedule 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 $\mathcal{R}$-schedule of a graph $G = (V, E)$ is the graph $G' = (V', E')$ where the vertices of V' are the routes of the $\mathcal{R}$-schedule 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 1. *Problem PRA cannot be approximate within a factor $n^{1-o(1)}$ unless* P = 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 $u_0, u_1, \ldots, u_l$ 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.

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 $\mathcal{R}$-schedule has a solution modulo P if and only if the $\mathcal{R}$-schedule 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 2. *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 $v_1, \ldots, v_l$ 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 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 $\mathcal{F}$ 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 $\mathcal{R}$-schedule 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 $\mathcal{R}$-schedule 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 $\mathcal{R}$-schedule. 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 $\mathcal{R}$-schedule 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 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 $\mathcal{R}$-schedule 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 $\mathcal{R}$-schedule 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 $\mathcal{R}$-schedule in $G = (V, A)$.

5.4.1 The 1-arc collision PRA problem

In this restriction of PRA, we are given a $\mathcal{R}$-schedule 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

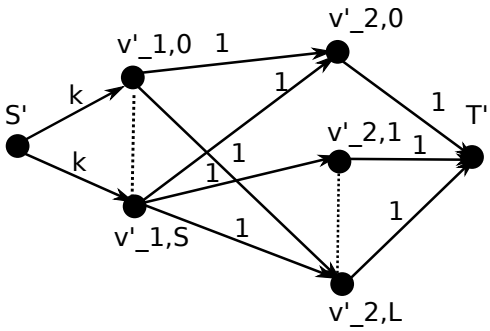

Figure 1: Reduction of DPA-NPA into a flow problem

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.