Greedy algorithms for scheduling periodic message

Maël Guiraud[1,2] and Yann Strozecki[1]

[1]David Laboratory, UVSQ
[2]Nokia Bell Labs France

***Abstract*—A recent trend in mobile networks is to centralize in distant data-centers processing units which were attached to antennas until now. The main challenge is to guarantee that the latency of the periodic messages sent from the antennas to their processing units and back, fulfills protocol time constraints. The problem is then to propose a sending scheme from the antennas to their processing units and back without contention and buffer.**

We study a star shaped topology, where all contentions are on a single arc shared by all antennas. We present several greedy heuristic to solve PAZL. We study their experimental efficiency and we use them to prove that when the load of the network is less than 44%, there is always a solution to PAZL. We also prove that for random lengths of the arcs, most of the instances have a solution when the load is less than 45%.

I. Introduction

II. Model

Describe the general model, and objective to optimize: no buffering. It is a scheduling/packing problem, could call each element a task or just an element. Notion of depth.

+ the simplified version for the star with a picture + a set of number $d_1, \dots, d_n$ and two values P, τ. Notion of load. [?]

III. Basic Algorithms

Notion of partial solution, how to extend it (and notation). All algorithms are greedy and most can work online. Once a route is placed, it is not changed, hence in a partial solution we are only interested in the set of positions used in the backward and forward period. If n tasks has been placed, we denote them by $x_1, \dots, x_n$ and $y_1, \dots, y_n$.

A. Depth one

Algorithm of increasing quality

First fit, analyzed naively $n/4$, then better through compacity $n/3$.

Meta interval, load of $n/3$.

First fit in order of the $d_i \mod \tau$ $n/2$, as good as naive first fit for $\tau = 1$ (all alg degenerate to this one for $\tau = 1$).

B. General graph

Use the coherent routing property. Algo first fit :$n/4$ for any τ and P, and any DAG of depth k.

Copy the results of the previous article and propose heuristics to chose among several positions/candidates (compacity heuristic).

C. Experimental results

Results better in practice: give the data Two heuristics to test:

- heuristic to build super compact assignment (among the compact assignments possible, chose the one which maximize the gain on the second bloc)
- heuristic to maximize the free position of the remaining elements

Quality of the results explained by the average analysis done later

IV. Why τ can be assumed to be one

Rank the d_i by value, and compute $d_i + d_n \mod \tau$. We allow buffering, but the worst time should not increase ! Bufferize each route during $\tau - (d_i + d_n \mod \tau)$. All routes have the same remainder mod τ, can assume they are of size one. A bit mor complex on the general graphs, proofs on the depth of the graph. Should take into account the length of the graph.

V. Above $1/2$

We show that we can go above $1/2$ of load using a two passes greedy algorithm. We use a two pass algorithm because it gives a better bound than a similar one pass greedy algorithm but mainly because it drastically simplifies the proof. We use the notion of potential of a route in a partial solution and we will show how to produce partial solutions with good potential, which can then be extended into a complete solution.

Definition 1. *The potential of a route of shift s in a partial solution (whether fixed or not in the partial solution), is the number of integers $i \in [P]$ such that i is used in the forward window and $i + s \mod P$ is used in the backward window. We denote by Pot(S) the sum of potentials of the routes in the partial solution S.*

Definition 2. *The potential of a position i in a partial solution is the number of routes of shift s such that $i + s$ is used in the partial solution.*

The potentials of the positions satisfy a simple invariant.

Lemma 1. *The sum of potentials of all positions in a partial solution of size k is nk.*

We then link Pot(S) to the potential of the positions in the forward window.

Lemma 2. *The sum of potentials of all used positions in the forward window in a partial solution S is equal to Pot(S).*

We now describe the algorithm to solve our problem with load $1/2 + \epsilon$. The first pass build a partial solution of size $P/2$ by chosing at each step the route and the position which increases Pot(S) the most. We partition the routes in three sets: R_1 the ϵP routes of largest potential in S, R_2 the routes placed in S and not in S_1 and R_3 the free routes not in R_1.

We consider the partial solution S' obtained by freeing the routes of R_1 in S. Then, S' is extended into a complete solution by using any greedy algorithm, such as first fit, using first the solutions of R_3 and finally R_1.

Lemma 3. *Let G be a weigthed bipartite graph G, with bipartition (A,B). If all vertices are of degree at least one then there is an edge of weigth at least the average of the weigths A plus the average of the weigths of B.*

Lemme faux, à mieux adapter au cas de la preuve. Plutot écrire un lemme qui dit qu'on augmente le potentiel d'au moins blah !

Theorem 4. *The two-pass algorithm always solves positively our problem with load $1/2 + 1/18$, when all routes have a distinct shift.*

On peut etre plus clean en choisissant k^2n/P mais le calcul est plus compliqué je crois.

Proof. We prove by induction on the size k of the partial solution S_k in the the first pass that $Pot(S_k) \geq (k-1)^2n/P$. The property is true for $k = 1$. Assume now it is true for k and we prove it for $k + 1$. First, remark that extending S_k into S_{k+1} add to $Pot(S_k)$ the potential of the position in the forward and backward windows used by the new route plus one.

First, we can assume that $Pot(S_k) < k^2n/P$, otherwise the property is proved since Pot is increasing with k. Hence by Lemma 2, the sum of the potential of the used positions in S_k is less than k^2n/P. By Lemma 1, the sum of potentials of the free positions in the forward window is at least $kn - k^2n/P = kn(1 - k/P)$. The average of potentials of the free forward positions is thus at least kn/P. The same argument can be made for free position of the backward window.

Because all routes have distinct shifts there is a least one route which can be placed in any position of the forward and backward position. Hence, there is a route which adds to the potential at least $2kn/P$ by Lemma 3 plus one because of itself. The algorithm selects the route which increases the potential the most, then $Pot(S_{k+1}) \geq 2k + 1 + Pot(S_k) \geq k^2n$, which proves the induction.

At the end of the first pass, we have a potential of $(P/2 - 1)^2n/P$, since $k = P/2$. The potential of a single route is bounded by $P/2$ since each placed route contribute at most one to its potential. We want to compute the minimum value m of the ϵP largest potentials. The case of minimum less than m but maximal potential corresponds to $\epsilon P - 1$ routes of potential $P/2$ and all other routes of potential $m - 1$. Hence,

$$(P/2 - 1)^2n/P \geq (\epsilon P - 1)P/2 + m(n - \epsilon P + 1)$$

.

We want m to be $4\epsilon P$, hence we must satisfy the following equation: (je vire un 1 qui est pénible et qui peut être optimisé avant).

$$nP^2/4 \geq 5/2\epsilon P^2$$

$$1/2 + \epsilon \geq 10\epsilon$$

$$\epsilon \leq 1/18$$

□

TODO: que peut-on dire quand il n'y a pas que des shifts distincts. Si on réapplique plsieurs fois la méthode, on obtient mieux, peut-être compliqué à calculer. On a pas besoin que tous les gars choisit soient à 2ϵ, seulement qu'ils soient à $2, \dots, 2\epsilon$. Est-ce que ça aide ? Je pense que oui et qu'on peut montrer qu'on a besoin que de la moitié du potentiel, petit lemme combinatoire supplémentaire qui donnerait $\epsilon = 1/10$.

VI. Algorithms for random instances

a) $\tau = 1$: We analyze the following process, called **Uniform Greedy** or UG. For each element in order, chose one admissible position uniformly at random. We analyze the probability that Uniform Greedy solve the problem, averaged over all possible instances. It turns out that this probability, for a fixed load strictly less than one goes to zero when m grows.

Définir l'ensemble des solutions de taille n parmi m.

Theorem 5. *Given an instance of size n uniformly at random UG produces a solution uniformly at random or fail.*

Proof. Regarder mes notes partielles pour compléter ça. □

Let us denote by $P(m,n)$ the probability that UG fails at the n_th steps assuming it has not failed before.

Theorem 6. *We have*

$$P(m,n) = \frac{\binom{n}{2n-m}}{\binom{m}{n}}.$$

In particular, $P(m,n) \leq f(\lambda)^m$, where $f(\lambda) < 1$.

Proof. Probability independent of the shift of the n element, can say it is 0. It is the probability that two sets of size n in $[m]$ are of union $[m]$. It is the same as the probability that it contains a given set of size $m-n$. Could find an asymptotic online. $\square$

Can we make the same argument for a deterministic algorithm? The not average version of the argument is the previous proof.

VII. Greedy and delay

Tradeoff between waiting time and load. Can we prove $0.5+\epsilon$ load for $f(\epsilon,n,\tau)$ waiting time ? No idea yet.

VIII. Non greedy algorithm

Greedy + swap one element if necessary. Can we guarantee a solution for a larger load ? Conjecture, yes for $\lambda = 2/3$. Seems hard for group theoritic reason (how to avoid subgroups of Z/pZ which are a problem)

IX. Lower bounds

Example/family of examples for which some greedy alg fail. Example/family of examples with a given load such that there are no feasible solution.

X. NP-hardness

Are we able to prove NP-hardness ?