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

There are easier cases, all routes are distinct or all routes are the same. There is a solution for load 1 in these two cases. With two distinct values, solution for load almost one. Insert a figure here, showing solutions for these cases. We give a method which always finds a solution for load $1/2 + \epsilon$.

To go above $1/2$ of load, we use a two-pass algorithm. In the second pass, a greedy algorithm is used to place a subsets of the routes, selected because they have less conflicts with the already placed routes. The first pass is not greedy, since we allow to change fixed route, but we could us a greedy first pass or even a single pass greedy algorithm to go over $1/2$. However, the proofs are much harder and the ϵ for which they hold is much smaller.

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 of the forward window, for 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 the following simple invariant.

Lemma 1. *The sum of potentials of all positions of the forward window 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 assign routes in any greedy manner, until it cannot assign some route anymore. Then, it applies some procedure described later which remove a route from the solution and add anther one. If at some point this procedure fails, it stops. When this algorithm stops, it can be shown that the potential of the obtained partial solution is larger than some value. Then, we select R the set of the ϵP routes of largest potential. The routes in R and in the partial solution are removed, then the free routes not in R are added to the partial solution and finally the route in R are added, using any greedy algorithm.

A. Swap and potential improvement

Let S be some partial solution of size k and let r be a free route of shift s. Assume that r cannot be used to extend S. The swap operation is the following: select a free position p, remove the route of position $p + s$ in the forward window of S and add r at position p in the forward window. We denote this operation by $Swap(r,p,S)$.

Lemma 3. *Let S be some partial solution of size k and let r be a free route of shift s. If r cannot be used to extend S, then either $Pot(Swap(r,p,S)) > Pot(S)$ or $Pot(S) \geq kn/2$.*

Proof. The positions in the forward window can be partitionned into two part: P_u the positions used in the forward windows and P_f the positions unused in the forward windows. Let us denote by V_f the value of the positions in P_f and by V_u the potential of the positions of P_u. By Lemma 2, since P_f and P_u partitions the positions, we have $V_f + V_u = kn$.

By hypothesis, since r cannot be placed, for all $p \in P_f$, $p+s$ is used in the backward window. We now define a function F which associates to $p \in P_f$ the position p' such that there is a route r' in S placed at p' in the forward window and at $p+s$ in the backward window. The function F is an injection from P_f to P_u. Rmeark now that if we compare $Swap(r,p,S)$ to S, on the backward window nothing changes. Hence the potential of each position in the forward window is the same. Hence, doing the operation $Swap(r,p,S)$ add to $Pot(S)$ the potential of the position p and removes the potential of position $F(p)$. Assume now, to prove our lemma, that for all p, $Pot(Swap(r,p,S)) \leq Pot(S)$. It implies that $V_f \leq V'_u \leq V_u$ and by Lemma 1 we have $V_f \leq Pot(S)$. Since $V_f + Pot(S) = kn$, we have that $Pot(S) \geq kn/2$. □

B. Analysis of the Algorithm

We give an analysis of the algorithm, showing that it works for some value of ϵ. We will later show that some refinments of this algorithm: a better selection of the values added in the second step, the possibility to repeat the first step to guarantee a higher potential yields a better ϵ.

Theorem 4. *The two-pass algorithm solves positively our problem with load $1/2 + 1/16$.*

Proof. The first pass of the algorithm guarantes that we obtain a partial solution S of size k such that $Pot(S) \geq kn/2$ by Lemma **??**. Moreover, $k \geq P/2$ since one can always place $P/2$ routes with a greedy algorithm.

At the end of the first pass, we have a potential of at least $Pn/4$ and we select the ϵP routes of largest potential. They must be of potential at least $2\epsilon P, 2\epsilon P + 2, \ldots, 4\epsilon P$. Sort all routes by decreasing potential and assume that the previous condition is not met, that is the ith route in order of potential is of potential less than $4\epsilon P - 2i$. The potential of a single route is bounded by $P/2$ since each placed route contribute at most one to its potential. Therefore, the first i routes are of potential at most $P/2$ and the following ones of potential at most $4\epsilon P - 2i$. Therefore the potential is less than $iP/2 + (4\epsilon P - 2i)(n - i)$. This function is decreasing for $i \leq \epsilon P$, hence the potential should be less than $4\epsilon Pn$.

For the algorithm to succeed, we want the potential to be larger than $4\epsilon Pn$ so that the condition on the routes of largest potential is met. Hence we must satisfy the following equation:

$$Pn/4 \geq 4\epsilon Pn.$$

$$\epsilon \leq 1/16.$$

□

Pour améliorer les résultats on peut répéter l'algo de swap une fois qu'on a obtenu au moins $(1/2 + \epsilon)P$ routes placées, pour obtenir $n^2/2$ en potentiel. On doit alors avoir $n^2/2 \geq 4\epsilon Pn$, ce qui donne $\epsilon \leq 1/14$. Si on veut pousser la technique plus loin, il faut améliorer la borne sur le potentiel. Au lieu de prendre les ϵP plus grandes routes, on prend pour i de 1 à ϵP la route de plus petit potentiel supérieur à $2\epsilon P$ et pour qu'elle existe, il suffit que le potentiel soit supérieur à $2\epsilon P(n - i)$. À vérifier, si à un moment tout est de potentiel $2\epsilon P$ au moins, alors c'est facile de conclure. On obtient alors $\epsilon \leq 1/6$, ce qui donne un algo dès que la charge est inférieure à $2/3$.

Autre possibilité, améliorer le potentiel en obtenant plus que la moyenne, en jouant notemment sur la symétrie Backward et Forward.

Expliquer les nombreuses raisons pourquoi ça marche mieux en pratique.

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. □

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. It can be used as the second step of the algorithm using potential. One can show that we use only the free routes and we swap if necessary. We show that, if the potential is at least $2\epsilon Pn$, then at least one of any $P - n$ elements has positive potential and can be moved. Such an operation lose at most $2n$ of potential. As a consequence, placing the last elements lose $2n\epsilon P$ in potential. Hence if the potential is more than $4\epsilon Pn$, the lagorithms terminates.

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 ?