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.

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

III. Basic Algorithms

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 Explanation by number of routes which can be dealt with high probability ????

heuristic to build super compact assignment (among the compact assignments possible, chose the one which maximize the gain on the second bloc)

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 $1/2$ using a greedy algorithm.

TODO: properly define the potential of a position, of a route. Relates both through a lemma showing that the sum of both is equal. Also a lemma which relates potential to the number of conflicts.

We assume that all distances are different (if at least a constant fraction are, it is ok). The greedy algorithm is as follows, for $(1/2 + \epsilon)P$ routes:

Let S be the set of routes of potential $2\epsilon n$, intialized to $\emptyset$. When a route is added to S, it is removed from the routes used to compute the potential. When $|S| \geq \epsilon n$, the algorithm stops. For $k < P/4$ routes already placed, the potential of a position i on the backward windows is the number of yet unused routes of delay d such that $i - d \mod P$ is used. This potential corresponds to the number of routes for which placing a route at this position remove only one possible offset instead of two. On the backward windows select the unused position of largest potential and find a route which can be placed at this position, not in the best $\epsilon n - |S|$ routes.

Proof it works:

Since all distances are differents, there are at least $(1/2 + \epsilon)n - |S| - k$ positions in the forward windows which allows to place a route attaining a given position in the backward windows. Since there are k used offsets in the forward windows, there are at least $(1/2+\epsilon)n - |S| - 2k$ free positions among them. Since $k < n/4$, there are at least $\epsilon n - |S|$ possible routes satisfying the constraint, hence one is not among the $\epsilon n - |S|$ routes having best potential (with the chosen variant, not useful to spare the best ones, simplifiyng the proof and the algorithm).

Assume that there is a free position, wich increases the global potential by at least $2/3$ of the average which is $k(1/2-k)/n$. Then the global potential is a least $(2/3)\sum_{i=1}^{k} i(1/2-i)/n$. On the other hand, if no position of at least $2/3$ the average is avalaible, then the global potential is at least $1/3k(1/2-k)$. (to clean that, prove by induction that the potential is always larger than some value at step k). If we can prove that when $k = P/4$, the potential is at least $2\epsilon n(1/2+\epsilon)n$, then it implies that the algorithm has stopped before that point and it implies there is a solution. We solve the equation lower bound on the potential larger than $2\epsilon n(1/2+\epsilon)n$ to obtain a correct value on ϵ.

The computation is not optimised at all. We win only on the backward windows but we should win as much on the forward windows. Several simplification in the bound (but of little impact). Lose too much when dealing with only positions of potential lower than the average (can do better than $1/3$ vs $2/3$).

VI. Algorithms for random instances

a) $\tau = 1$: Use the same argument as in the previous proof: if there are n routes placed, then chosing randomly a new one it has $2n$ constraints. However some constraint are redondant: (x_i) position in the forward windows and (y_i) in the backward. Compute the probability that for a randomly chosen d_{n+1} and $|\{i \mid \exists j x_i + d_{n+1} = y_j\}| > k$.

If we compute the expectation of this set: $n/2$, with $P = 2n$ (we can show such a result with high probability, using chernov). The new routes placed cost two constraints, hence we can expect to place around $n + \lambda n$ routes such that $2(n+\lambda n) - n/2 < 2n$ hence $\lambda = 1/4$. Hence the "average" constant is $5/8$

If we can somehow argue that the new routes placed are also randomly distributed, then $2\lambda n - \lambda n \lambda < n$.

$$2\lambda - \lambda^2 = 1$$

Too good to be true ? Il y a un problème.

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 ?