The Greedy Algorithm and its Application to the Construction of a Continuous Speech Database

Hélène François, Olivier Boëffard

IRISA - Institut de Recherche en Informatique et Systèmes Aléatoires
Université de Rennes 1, Enssat, Lannion, France
{francois, boeffard}@enssat.fr

Abstract

Databases containing varied linguistic features can be build by condensing large corpora; in this work we need to cover a set of phonetic units with a minimal set of natural phonetic sentences. With this aim in view we compare three set covering methods: the *greedy* method, its inverse which we call the *spitting* method, and the *pair exchange* method. Each method is defined with several criteria guiding the selection of sentences; they relate to the number of units of the sentences, to their length, and to the rareness of their units. A first experiment shows that pair exchange method doesn't guarantee a total covering. Greedy and spitting methods performances are comparable; nevertheless greedy is a bit better and above all less time-consuming. Applying spitting method to a greedy cover increases performance by removing about 10% redundancy. So does pair exchange method, but it is more time-consuming. Most of the criteria guiding selections are sensitive to the sentences length. Criteria performances obtained for a total covering are not necessarily transposable to a partial covering.

1. Introduction

This work[1] deals with the general framework of the construction of databases presenting varied linguistic features. We need a continuous speech database made of a maximum of phonetic units, but remaining small at the same time. After its recording, this set of sentences will constitute the source from which the syntheziser of our text-to-speech synthesis system will draw the acoustic units it needs. The problem is to find, among the sentences of a large database, a subset of sentences which covers all the units we need, this subset being as small as possible. Solutions to this problem can be applied in other fields of Language Ressources, for example to build evaluation databases for speech recognition or dialogue, where sentences - or paragraphs, or dialogue transcriptions - are searched in a large base to cover some characteristics to ensure diversity in the evaluation corpus.

We tackle this question as a set covering problem (SCP) (François and Boëffard, 2001). Since it is NP-hard, we must resort to heuristics to solve it on large databases. Here we study *greedy* methods. They consist in building the cover "step by step". A first sentence is selected according to a criterion; for example, the number of units to cover that it contains. The sentence is added to the cover, and the covered units are removed from the set of units to cover. The process starts again; the second sentence, in this example, contains a maximum of non-already-covered units. The process stops when all the units are covered. Performance of this method depends on the way sentences are chosen therefore on the *criterion* which guided their selection; several choices are conceivable.

Gauvain et al. (1990), who used the greedy method to build an evaluation base for speech recognition, organized criteria hierarchically. Sentences were first classified in accordance with their length in words, which were short or long. Then they had to contain a minimum number of new units, and short sentences had to have a maximum number of 5 punctuation marks. 11 000 sentences on 170 000 were selected that way, with the aim of maximizing the number of units types (triphones). This base was also used for speech synthesis (Prudon and d'Alessandro, 2001).

Black and Lenzo (2001) adapted the greedy algorithm to the future database organization, that is to decision trees which are searched through to find units to synthesize speech. The cover related to the tree nodes.

Kawai et al. (2000) resorted to a *pair exchange* method. At initial state, the cover contains a given number of sentences chosen arbitrarily. An out-of-cover sentence and an in-cover sentence are chosen. The two sentences are exchanged temporarily; if the covering is better, that is more units are covered and the cover is smaller, the change is kept, otherwise it is rejected. This process is repeated as many times as needed.

Rojc and Kačič (2000) combined a "reduction" approach with a pair exchange method to build a 1200 sentences triphones-rich database, with at least 10 representatives per triphone type. They defined 4 corpora of 5 000 sentences. At first each corpus was relieved of its useless sentences, which removal did not damage the covering; this process stopped either when 1200 sentences remained or when removal was not possible anymore. This first phase is a kind of "inverse greedy" which spits useless sentences instead of eating useful sentences. Subsequently the poorest sentences of the richest corpus - called the target corpus - were pair exchanged with the richest sentences of the 3 other corpora if it improved the covering, so that to enrich the target corpus.

What is at stake in the work presented here is the comparative evaluation of the three methods used in that works, that is to say the *greedy* method, its inverse, we will call it the *spitting* method, and the *pair exchange* method; they are introduced in section 2. The performances of these methods depend on the criteria selecting sentences one after another; they are detailed in section 3. Results are exposed

[1]This work is financed by France Telecom R&D within contract with DIH/IPS/VMI laboratory.

in section 5., their interpretation in section 6.

2. Greedy, spitting and pair exchange methods

Greedy methods are a simple heuristic solution to the set covering problem, they can be adapted to many applications. The cover grows step by step, each stage taking the results of the previous stage into account.

General rules governing the three greedy, spitting and pair exchange methods are presented in fig. 1. In each case, a modification is tried on the current cover - an addition, a removal or an exchange of sentences according to the method. If the modification is fruitful, it is kept, otherwise another one is attempted.

- **Initialize the cover C:**
- Greedy: empty cover
- Spitting: all-sentences cover
- Exchange: given cover
- **Define the space of candidates S**
- Greedy: out-of-cover sentences
- Spitting: in-cover sentences
- Exchange: out-of-cover sentences, an in-cover sentence s^{test} being tested
- **While stop condition is false**
- Greedy: all units are covered
- Spitting: another removal would damage the covering
- Exchange: pair exchange is not possible anymore
  - **Look for the best change according to criteria**
  - Greedy: the most useful sentence
  - Spitting: the most useless sentence
  - Exchange: the most profitable exchange
    - **For each criterion k, reduce S**
      - **Assign a weight to each sentence**
      - **Look for the best weight**
      - **Re-assign S with the best-weighted sentences**
    - **Extract 1 sentence s^{best} from the last S**
  - **If change is possible, do it:**
  - Greedy: add s^{best} to C, remove it from S
  - Spitting: remove s^{best} from C, add it to S
  - Exchange: exchange s^{test} and s^{best}
  - **Update stop condition**
- **End.**

Figure 1: Generic algorithm. Each step is instantiated for the greedy, the spitting and the pair exchange algorithms.

The covering may be wished *total* (100% units are covered), or *partial*, in this case the threshold is rather expressed by the accepted cover size, either in sentences or in units instances.

The greedy algorithm begins with an empty initial cover. The first chosen sentences constitute a partial covering which increases. Total covering is achieved at the end of the algorithm. Algorithm continuation would only bring redundancy.

The spitting algorithm begins with a "full" cover, that is the whole set of sentences; covering is thus total, the algorithm does not have to look for missing units. Sentences are removed one by one until no sentence could be removed without damaging the total covering. Algorithm continuation produces a partial covering.

The pair exchange algorithm is a bit different from the others because its aim is more to improve the cover than to build it, either by increasing the number of covered units at a cover size threshold, or by decreasing the cover size at a threshold of covered units. The number of sentences is invariant.

The algorithms efficiency depends on the criteria used to select sentences one by one. The following section presents different options.

3. Criteria for sentence selection

For the three algorithms, candidate sentences are characterized in accordance with the modification that would be produced by their additon, removal or exchange. A cover can be characterized by several parameters as shown in fig. 2. Among the units present in this schematic cover, some of them are *useful units*, that is the units to cover, some are *useless units*, that is units not to cover but collected *en passant*. The instances of useless units are of course useless to the covering. On the other hand, among instances of useful units, some of them will be counted as *useful instances*, and other instances as *useless instances*; if we need 5 representatives of a unit type and we have 9 of them, we count 5 useful and 4 useless instances.

Criteria presented here relate to the number of units of the sentence and to the number of useful instances either of the sentence or of the cover; distinction between units types and units instances in a sentence for criteria makes sens when several representatives of a unit type are needed, otherwise it comes to the same thing. Criteria relate to the presence of rare units as well; there is a big difference between the observed units distribution and the wished quasi-uniform distribution, that's why covering rare units is costly; it may be of interest to treat them first. Besides the number of covered units, the sentences length and the cover cost are as well taken into account.

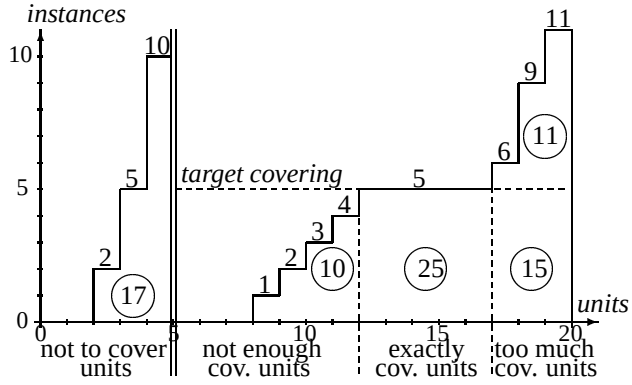

Figure 2: *Cover characterization. In this partial covering scheme where 5 representatives per unit are needed, there are 3 useless units (left part) and 12 useful units (right part) in the cover. As for units instances, counted in the circles, 10+25+15 are useful and 17+11 are useless.*

3.1. Criteria for the greedy algorithm

The greedy algorithm begins with an empty initial cover. We used the following criteria:

- **1g:h-uus,l-sc** : among the sentences having the **h**ighest number of **u**seful **u**nits in the **s**entence, the chosen sentence is the one with the **l**owest **s**entence **c**ost;
- **2g:h-uus-on-sc** : the chosen sentence has the **h**ighest number of **u**seful **u**nits in the **s**entence weighted by the **s**entence **c**ost;
- **3g:h-uis-on-sc** : the chosen sentence has the **h**ighest number of **u**seful **i**nstances in the **s**entence weighted by the **s**entence **c**ost;
- **4g:ra,h-uus-on-sc** : among the sentences containing the **ra**rest non-already-covered units, the chosen sentence is the one with the best h-uus-on-sc score.
- **5g:ra,l-sc** : among the sentences containing the **ra**rest non-already-covered units, the chosen sentence is the one with the **l**owest **s**entence **c**ost.

3.2. Criteria for the spitting algorithm

The spitting algorithm is able to begin either with a "full" cover, that is to say the whole set of sentences, or with the cover obtained with the greedy algorithm; the covering will be relieved of its useless sentences. We used the following criteria:

- **1s:lb-uic** : the chosen sentence respects a given **l**ower **b**ound for the number of **u**seful **i**nstances in the **c**over; the bound here corresponds to the total cover, so this criterion means that the sentence is useless;
- **2s:lb-uic,h-sc** : among sentences verifying lb-uic, the chosen sentence is the one having the **h**ighest **s**entence **c**ost in instances;
- **3s:lb-uic,h-lrai** : sentences verifying lb-uic are assigned with a weight which is the number of **i**nstances of their **ra**rest unit; the chosen sentence is the one which has the **h**ighest value. The aim is to avoid to remove sentences containing the rarest units, so that other sentences can be removed after them.

3.3. Criteria for the pair exchange algorithm

The pair exchange algorithm is able to begin either with a total covering obtained with the greedy or the spitting algorithms or their combination, or with a set of sentences chosen arbitrarily. We used the following criteria:

- **1e:lb-cuc,d-cc** : among exchanges preserving a given **l**ower **b**ound for the number of **c**overed **u**nits in the **c**over, the chosen exchange is the one which **d**ecreases the **c**over **c**ost.
- **2e:i-cuc,d-cc** : among exchanges **i**ncreasing the number of **c**overed **u**nits in the **c**over, the chosen exchange is the one which **d**ecreases the **c**over **c**ost at the same time.

4. Experiments

This section describes the evaluation database and introduces the three experiments which were carried out.

4.1. Database

The evaluation database was designed so that results could be transposable to a larger database, which is the "Irisa" database we presented in (François and Boëffard, 2001). That evaluation base had to be able to cover 95% of the units instances, units being diphones. It consists of 3 000 sentences issued from interviews of comic-strip authors. It contains 1 037 diphones types. The 493 (48%) most frequent diphones allow 95% of the instances to be covered. The rarest units have 52 instances. We need 3 representatives at least per unit, total covering were thus possible.

4.2. Three experiments

First stage was the comparative evaluation of the methods presented in section 3. used *alone*. It compares the criteria of the greedy algorithm ("1g" to "5g"), the criteria of the spitting algorithm beginning with a full cover ("1s" to "3s") and the criterion "1e" for the pair exchange algorithm beginning with an arbitrary cover containing the first 188 sentences of the evaluation base.

Second stage consisted in *combining* algorithms, first by appending the spitting algorithm to the greedy algorithm, then by applying the pair exchange algorithm to the three following cases: the best greedy cover, the best spitting cover and the best cover issued from the "greedy then spitting" combination.

The last stage concerned a dynamic analysis used to predict partial covering results.

5. Results

Here are the results for the three experiments presented in the previous section: algorithms used alone, algorihms combined, and dynamic behaviour of the greedy algorithm.

Let's have a rough idea of the absolute values first. There are 493 units to be covered 3 times at least, that is $n = 1479$ instances are to be covered. An "ideal" cover, in which every unit would be covered 3 times exactly, would then have a cost (its size) of C_{ideal} = 1479 instances. Maximum performance ratio r^{greedy} was established more and more precisely for the simple unweighted greedy algorithm (h-uis criterion). $r^{greedy} = C_{max}/C_{best}$, where C_{max} is the cost upper bound and C_{best} the lowest cost, that is the one we would like to achieve. Table 1. give these ratios; they are $H(n)$ (Lovasz, 1975), $ln(n)$ (Feige, 1996), and $ln(n) - ln(ln(n))$ (Slavìk, 1996). It also provides a low approximation $C_{max,approx.}$ of C_{max} by taking C_{ideal} instead of C_{best}.

5.1. Algorithms used alone.

Table 2. shows the first stage results. The initial base contains 3 000 sentences and 157 265 instances thus 52.4 inst./sent. on average. This table presents the scores of covers obtained randomly (frame A), with the greedy method (frame B), with the spitting method (frame C) and with

Performance ratio - Formula	Perf. ratio - $n = 1479$	$C_{max,approx.}$ ($C_{best} = 1479$)
$H(n) = \sum_{i=1}^{n} 1/i$	7.8767	11650
$ln(n)$	7.2991	10796
$ln(n) - ln(ln(n))$	5.3114	7856

Table 1: Upper bounds of the greedy algorithm for the basic unweighted case.

CRITERIA	number of sentences in cover	nb of inst. in cover	ave. sent. lenght (inst./sent.)
Initial base	3000	157265	52.4
A. Random selection ('r')			
1r:fi rst 188 sent.	(29nu) 188	11483	61.0
B. Greedy algorithm ('g'), initial cover is empty			
1g:h-uus,l-sc	**48**	6543	136.3
2g:h-uus-on-sc	231	**4445**	19.2
3g:h-uis-on-sc	216	**4306**	19.9
4g:ra,h-uus-on-sc	137	**4312**	31.5
5g:ra,l-sc	304	5299	17.4
C. Spitting algorithm ('s'), initial cover is full			
1s:lb-uic	113(-2887)	7776	68.8
2s:lb-uic,h-sc	264(-2736)	**4649**	17.6
3s:lb-uic,h-lrai	**89**(-2911)	8289	93.1
D. Pair exchange ('e'), initial cover = fi rst 188 sentences			
1e:lb-cuc,d-cc	-	-	-
2e:i-cuc,d-cc	(4nu) 188	10780	57.3

Table 2: Condensation results according to criteria - greedy, spitting and pair exchange algorithms were used alone. Performance, related to cover size, is given in 2 ways: number of sentences and number of instances. "Xnu" in cases 1r and 2e mean that there are X **n**ot covered **u**nits: total covering was not achieved in that cases. Boldface values correspond to the best cases of total covering, either according to the number of sentences or according to the number of instances. Criterion 1e has no value because it can't be used alone, thus not in this phase. Negative values in spitting cases are the number of rejected sentences.

the pair exchange method (frame D). The average length of sentences appears as well.

The random covering is partial (94% units are covered 3 times at least), its cost is of 11 483 instances.

The greedy method (frame B) produces total covering. Best results in instances are achieved in cases "2g", "3g" and "4g", best results in sentences in cases "1g". Nevertheless cases "2g", "3g" and "4g" have not the same scores in sentences: the "4g" case has got less sentences than "2g" and "3g"; they are longer (31.5 inst. on ave., vs. 19 for "2g" or "3g"). Case "1g" is the best in sentences and the worst in instances at the same time: this cover contains very long sentences (136.3 inst. on ave.). The "5g" criterion produces the shortest sentences (17.4 inst. on ave.). All scores are under the lowest threshold of 7 856 instances presented in table 1.

The spitting method (frame C) produces total covering as well. Best results occure in case "2s" in instances, and in case "3s" in sentences, but in this case the cost in instances is almost twice the "1g" cost (8289 vs. 4649 instances).

The pair exchange algorithm (frame D) is only presented in case "2e"; case "1e" needs a total cover at its start. "2e" produces a partial covering: 4 units were not covered. It also contains a high number of instances, which is almost the size of the random "1r" case (10 780 vs. 11 483 instances).

This analysis shows that in the studied case, only greedy and spitting algorithms guarantee total covering. Their results are comparable although greedy is a bit better. Another interesting aspect is that spitting is more time-consuming; it has to remove 2 736 sentences when greedy has to take only 216 sentences. These results are discussed in section 6.

5.2. Algorithms combined

CRITERIA	number of sentences in cover	number of instances in cover	ave. sent. lenght (inst./sent.)
E. Greedy algorithm, then spitting algorithm			
1g,1s	**48**(-0)	6543	136.3
1g,2s	**48**(-0)	6543	136.3
1g,3s	**48**(-0)	6543	136.3
2g,1s	188(-43)	**4017**	21.4
2g,2s	188(-43)	**3966**	21.1
2g,3s	187(-44)	**4026**	21.5
3g,1s	182(-34)	**4009**	22.0
3g,2s	183(-33)	**4004**	21.9
3g,3s	182(-34)	**4009**	22.0
4g,1s	**129** (-8)	4195	32.5
4g,2s	**129** (-8)	4141	32.1
4g,3s	**129** (-8)	4174	32.4
5g,1s	261(-43)	4737	18.1
5g,2s	263(-41)	4740	18.0
5g,3s	262(-42)	4767	18.2
F. Greedy algorithm, then pair exchange algorithm			
3g,1e	216	4051	18.8
G. Spitting algorithm, then pair exchange algorithm			
2s,1e	264	4649	17.6
H. Greedy, then spitting, then pair exchange algorithms			
2g,2s,1e	188	**3872**	20.6

Table 3: Condensation results according to criteria. - greedy, spitting and pair exchange algorithms combined. Performance, related to cover size, is given in 2 ways: number of sentences and number of instances. Boldface values correspond to the best cases of total covering, either according to the number of sentences or according to the number of instances. Negative values in spitting cases are the number of rejected sentences.

Results of the second experiment are shown in table 3; greedy, spitting and pair exchange algorithm are combined each after another. Frame E gives the score of the spitting algorithm applied to greedy-obtained covers, frame F,

G and H give the scores of the pair exchange method applied to varied cases.

Applying the spitting algorithm to the covers obtained with the greedy algorithm (frame E) brings about an improvement of (-18.6% sentences, -10.8% instances) in case "2g,2s", (-15.3% sent., -7.0% inst.) in case "3g,2s", and (-5.8% sent., -4.0% inst.) in case "4g,2s", which is of interest. Besides general behaviours remain: the three best scores in instances are still the improved cases "2g,*s", "3g,*s" and "4g,*s" for the greedy algorithm, and the cases "*g,2s" for the spitting algorithm.The spitting algorithm reject sentences which became useless after their selection by the greedy algorithm.

The pair exchange algorithm (frames F, G, H) provides an improvement in case "3g,1e" (-5.9% instances) and in case "2g,2s,1e" (-2.4% instances), but not in case "2s,1e". Improvements of case "3g" produced by spitting (case "3g,2s") or pair exchange (case "3g,1e") are comparable, nevertheless spitting is less time-comsuming, it only removes 33 sentences while pair exchange tests every of the 216 sentences several times.

The lowest of the lowest scores in instances goes to the case "2g,2s,1e" which cumulates the 3 methods; improvements are of 7.0% in instances with the spitting algorithm and of 2.4% with the pair exchange method, that is a total improvement of almost 10% in instances. These results are discussed in section 6.

5.3. Dynamic study of the greedy algorithm

The analysis of the cover cost progress according to the number of covered units gives the results we would have for a partial covering (fi g. 3). For example, if we consider a greedy covering stopped at 2 000 instances, the number of covered units varies from 50 to 70% according to the criterion used. Methods having similar scores for total covering ("2g", "3g", "4g") have not necessarily the same score for a partial covering: "4g" covers 62% of the units while "2g" and "3g" cover 70% of them.

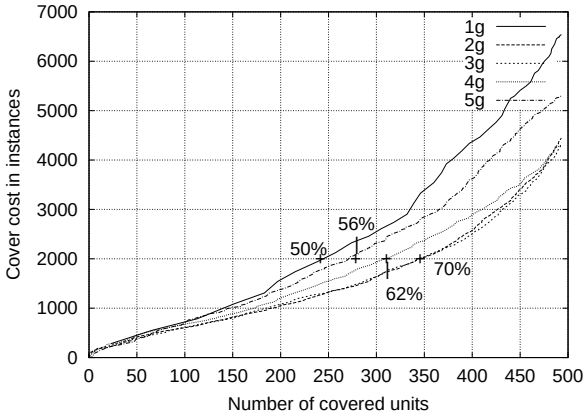

Figure 3: Cover cost vs. number of covered units. There are 493 units.

Figure 4 shows the value of the "3g" criterion along selections, that is to say the number of useful instances of the selected sentence weighted by its cost, which is its length in instances. We see that after the 73th sentence, sentences bring more useless units than useful units, thus they contribute rather to the cost increase than to the covering increase.

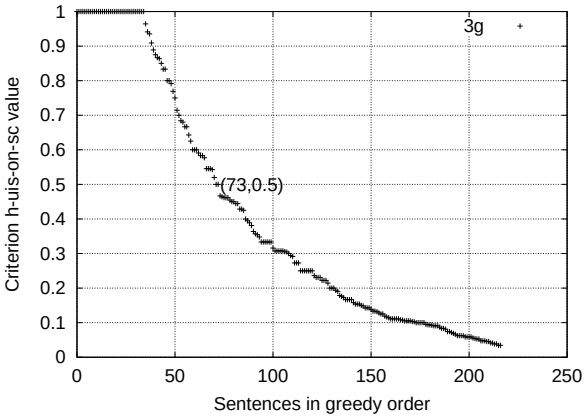

Figure 4: Number of useful instances in the sentence weighted by its cost (its total number of instances). After the 73th sentence, sentences bring more useless than useful instances.

6. Discussion

The above results are commented in this section.

6.1. Algorithms used alone

The following considerations are associated with table 2.

Criteria organization: hierarchic or simultaneous ? Criteria "1g" and "2g" (table 5.1.) both involve the number of useful units per sentence (h-uus) and the sentence cost (sc) in instances. Criteria "1g" considers them *hierarchically*: fi rst sentences are extracted with "h-uus", then among them are extracted those with the least cost. Criteria "2g" uses them *simultaneously* by considering their ratio. Results are really different both for the sentences number and for the instances number. In case "1g", criterion "h-uus" has much more influence than "sc", while in the "2g" case they are more balanced. Criteria organization can change results signifi cantly.

A maximum of units types or of units instances ? Criteria "2g" and "3g" (table 5.1.) are close in their design and results, aside from the fact that criterion "2g" is interested in the maximum weighted number of useful *units* while criterion "3g" works with the maximum weighted number of useful *instances*. "3g" is a bit better than "2g", but we consider this difference not signifi cant, all the more so because this difference may reduce with triphones, for which each unit has most of time only one instance per unit in a sentence; so the number of units equals the number of instances. It would probably not be so for phones. We chose criterion "2g" which is less time-consuming, nevertheless "3g" makes more sens because it realises a "standard" weighting, which is the ratio between the number of useful instances on the total number of instances of the sentence.

Length of the selected sentences. Criteria have a influence on the length of the chosen sentences. The evaluation

base has 52.4 instances per sentence on average (table 5.1.); criterion "2s" brings about sentences with 17.6 instances on ave., and criterion "1g" an average of 136.3 instances per sentences, so variance is substantial. The most length-independent criteria are cases "1s", where no units count intervene directly, and "4g". A way of weakening the criteria influence on sentences length is to look straightaway for several kinds of sentences, short and long, as Gauvain et al. (1990) did.

Criterion 3s disappointment. The spitting algorithm is interested in useless sentences, because they only bring redundancy. But how to choose, among these sentences, the one which have to be removed first to have a low cover cost at the end? Criterion "2s" consists in removing first the long sentences; it is the best among the 3 criteria. Being removed at the beginning of the algorithm may be their only chance to be rejected, otherwise they may fast become essential because they have many units types compared to short sentences. Criterion "3s" removes among the useless sentences the one which rarest unit in the base is the least rare in comparison with the other useless sentences. The governing idea is to protect the scores of the rarest units so that they don't become essential, because so they would make essential the sentences they belong to. Nevertheless criterion 3s proved to be inefficient.

6.2. Algorithms combined

The following considerations come from table 3.

Is it worth applying the spitting after the greedy algortihm ? The spitting algorithm removes up to 18.6% sentences and 10.8% instances. It does it rapidly, for us its utility is blatant. Besides it heightens a bit the average sentence length, it removes more short than long sentences.

Does the spitting algorithm fit better with some of the greedy criteria ? We can't really bring an answer to this question. "2g,2s" combination is more efficient than "3g,2s" while "3g" case was a bit better than "2g", but differences are so weak that we consider them not significant.

Is it worth applying the pair exchange algorithm ? In the studied case the pair exchange algorithm proved to be of small interest, being too time-consuming, even if it helps to reduce the cover cost.

6.3. Dynamic study of the greedy algorithm

Which greedy criterion is performant with a partial covering aim ? Results presented in fig. 3 show that the performance in instances achieved with a total covering aim are not necessarily the same as those obtained with a partial covering goal. Cases "2g", "3g" and "4g" are equivalent in instances at the end of the total covering, but "2g" and "3g" are better if the algorithm is stopped before this moment. Besides the "4g" case priviledges the cover of rare units, thus its set of covered units would be probably very different from those obtained with criterion "2g" or "3g", which would probably have rather the frequent types of units.

Where should partial covering be stopped ? Partial covering are often stopped at a cover cost threshold either in sentences or in instances. Nevertheless it seems to be worth looking at criterion "h-uis-on-sc" (fig. 4) to take this decision, because the performance ratio between the number of brought useful instances and the cost increase decreases fast. This criterion can be a good tool to stop the algorithm.

7. Conclusion

Three covering methods were compared experimentally with an evaluation database of 3 000 sentences so that to cover 500 diphones, each being represented 3 times at least. These are the greedy method, the spitting method which is a sort of inverse greedy, and the pair exchange method. Several criteria were studied for each method; they depend on the number of useful units types or instances of the sentences, on their length, or on the presence of rare units therein.

In the studied case, total covering is guaranteed only by the greedy or spitting methods. These methods have comparable performance results as for the number of instances, the greedy being slightly better, but above all less time-consuming than the spitting method. The spitting method proved to be performant when applied after the greedy, removing up to 10% of the redundant instances. The pair exchange algorithm as well, with up to 6% instances removed, but it is more time-consuming. The greedy method followed by the spitting and the pair exchange method produce the best results in the studied case. Results obtained for a total covering are not necessarily transposable to partial covering.

An other stake for these covering methods is the combination of several criteria which may be of varied natures; van Santen and Buchsbaum (1997) propose a greedy method with sub-vectorization or based on models, and present an optimized version of it (Buchsbaum and van Santen, 1996). This was applied by Chu et al. (2001) to cover about 1 600 Mandarin tonal syllabes according to acoustic and prosodic criteria.

An interesting evolution would be to attempt to adapt to the studied case the GRASP, "greedy randomized adaptative search procedure" (Resende, 1998), the "randomized rounding technique" (Raghavan and Thompson, 1987; Srinivasan, 1995) and the "method of alteration" (Srinivasan, 2001).

8. References

A. W. Black and K. A. Lenzo. 2001. Optimal data selection for unit selection synthesis. In *Proceedings of the 4th ISCA Tutorial and Research Workshop on Speech Synthesis (SSW4)*, Perthshire, Scotland.

A. L. Buchsbaum and J. P. H. van Santen. 1996. Selecting training inputs via greedy rank covering. In *Proceedings of the 28th ACM Symposium on Theory Of Computing (STOC)*, pages 288–295, Philadelphia, PA, USA.

M. Chu, H. Peng, H. Yang, and E. Chang. 2001. Selecting non-uniform units from a very large corpus for concatenative speech synthesizer. In *Proceedings of the 26th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Salt Lake City, Utah, USA.

U. Feige. 1996. A threshold of ln n for approximating set cover. In *Proceedings of the 28th ACM Symposium on Theory Of Computing (STOC)*, pages 314–318, Philadelphia, PA, USA.

H. François and O. Boëffard. 2001. Design of an optimal continuous speech database for text-to-speech synthesis considered as a Set Covering Problem. In *Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, Aalborg, Denmark.

J.-L. Gauvain, L.F. Lamel, and M. Eskenazi. 1990. Design considerations and text selection for BREF, a large French read-speech corpus. In *Proceedings of the 1st International Conference of Spoken Language Processing (ICSLP)*, volume 2, pages 1097–1100, Kobe, Japan.

H. Kawai, S. Yamamoto, N. Higuchi, and T. Shimizu. 2000. A design method of speech corpus for text-to-speech synthesis taking account of prosody. In *Proceedings of the 6th International Conference of Spoken Language Processing (ICSLP)*, Beijing, China.

L. Lovasz. 1975. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13:383–390.

R. Prudon and C. d'Alessandro. 2001. A selection/concatenation text-to-speech synthesis system: databases development, system design, comparative evaluation. In *Proceedings of the 4th ISCA Tutorial and Research Workshop on Speech Synthesis (SSW4)*, Perthshire, Scotland.

P. Raghavan and C. D. Thompson. 1987. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374.

M. G. C. Resende. 1998. Greedy randomized adaptative search procedures (GRASP). Technical Report 98.41.1, AT&T Labs.

M. Rojc and Z. Kačič. 2000. Design of an optimal slovenian speech corpus for use in the concatenative speech synthesis system. In *Proceedings of the 2nd International Conference on Language Resources and Evaluatio (LREC)*, volume 1, pages 321–325, Athens, Greece.

P. Slavìk. 1996. A tight analysis of the greedy algorithm for set cover. In *Proceedings of the 28th ACM Symposium on Theory Of Computing (STOC)*, Philadelphia, PA, USA.

A. Srinivasan. 1995. Improved approximations of packing and covering problems. In *Proceedings of the 27th ACM Symposium on Theory Of Computing (STOC)*, pages 268–276, Las Vegas, NV, USA.

A. Srinivasan. 2001. New approaches to covering and packing problems. In *Proceedings of the 12th ACM-SIAM Symposium On Discrete Algorithms (SODA)*, pages 567–576, Washington, DC.

J. P. H. van Santen and A. L. Buchsbaum. 1997. Methods for optimal text selection. In *Proceedings of the 5th European Conference on Speech Communication and Technology (Eurospeech)*, pages 553–556, Rhodes, Greece.