Comparing set-covering strategies for optimal corpus design

Jonathan Chevelu, Nelly Barbot, Olivier Boeffard and Arnaud Delhay

IRISA - Institut de Recherche en Informatique et Systèmes Aléatoires
Université de Rennes 1, Enssat, Lannion, France
{Jonathan.Chevelu,Nelly.Barbot,Olivier.Boeffard, Arnaud.Delhay}@irisa.fr

Abstract

This article is interested in the problem of the linguistic content of a speech corpus. Depending on the target task, the phonological and linguistic content of the corpus is controlled by collecting a set of sentences which covers a preset description of phonological attributes under the constraint of an overall duration as small as possible. This goal is classically achieved by greedy algorithms which however do not guarantee the optimality of the desired cover. In recent works, a lagrangian-based algorithm, called *LamSCP*, has been used to extract coverings of diphonemes from a large corpus in French, giving better results than a greedy algorithm. We propose to keep comparing both algorithms in terms of the shortest duration, stability and robustness by achieving multi-represented diphoneme or triphoneme covering. These coverings correspond to very large scale optimization problems, from a corpus in English. For each experiment, *LamSCP* improves the greedy results from 3.9 to 9.7 percent.

1. Introduction

In automatic speech recognition as well as speech synthesis fields, many technologies rely on models trained on large speech corpora. The quality of these models depends strongly on the linguistic content of these corpora. So as to cover the maximum of the required descriptive attributes (mainly phonological and linguistic attributes), two strategies are conceivable. The first one consists in collecting randomly the acoustic materials. This strategy becomes quickly expensive because of the natural exponential distribution of the linguistic events. Unfortunately very few events take place very frequently compared with a considerable mass of rare events. This drawback becomes often acute owing to the need of many technologies to have several occurrences of a same event. Furthermore, this method does not guarantee the stability of the corpus content and its main characteristics as corpus size, sentence length, etc.. This situation may influence the learning of the model parameters.

One alternative consists in explicitly controlling the content of the learning corpus according to the target application. The main difficulty is to assure the presence of units longer than a phoneme, given their heavy-tailed distribution. A solution is the automatic extraction, from huge text corpora, of a subset which covers all the descriptive attributes and minimizes the speech duration after recording. This optimization problem can be translated as a *Set-Covering Problem* (*SCP*) that is a NP-hard problem. It is thus necessary to use sub-optimal or heuristic algorithms. Within the field of automatic speech processing, the most often used methodology is the greedy method based on an agglomeration policy. This iterative algorithm chooses at each step a sentence corresponding to the highest score which quantifies a sentence contribution to the iterated covering. (Gauvain et al., 1990) applies this greedy strategy to build a database for a speech recognition task thanks to hierarchically organized covering attributes. (Van Santen and Buchsbaum, 1997) studies several greedy variants of text selection, varying the required unit nature (diphone, duration components, etc.) and the sentence score function according to the application. In (François and Boëffard, 2001), the agglomeration greedy method is based on a heuristic which tries to satisfy the covering constraint with a priority on the rarest unit classes. This work has been recently implemented to build the Neologos corpus (Krstulovic et al., 2006). (Krul et al., 2006) builds a corpus whose diphoneme/triphoneme distribution approximates a uniform distribution. The greedy strategy is driven by a sentence cost function based on the Kullback-Liebler divergence. A similar method is used in (Krul et al., 2007) to construct a reduced database whose unit distribution is close to a given domain specific distribution. From an algorithmic point of view, (Kawai et al., 2000) proposes a pair-exchange mecanism. In (Rojc and Kacic, 2000), the first spitting greedy algorithm is introduced, that deletes uninteresting sentences, and is followed by a greedy pair exchange. In (François and Boëffard, 2002), several combinations of greedy algorithms -agglomeration, spitting and pair exchange- are studied and applied to the construction of a speech synthesis corpus. According to this work, the best combination is the agglomeration greedy algorithm followed by a spitting greedy algorithm. During the agglomeration greedy phase, the score of a sentence corresponds to the number of its units that are missing in the ongoing covering divided by the sentence length. As regards the spitting phase, at each step, the longest redundant sentence is excluded of the covering. In order to clarify the rest of our paper, this algorithm is called *ASA -Agglomeration and then Spitting Algorithm-*.

As an alternative to a greedy strategy, (Chevelu et al., 2007) proposes a solution based on the lagrangian relaxation. Indeed, solving a *SCP* by lagrangian relaxation may find an exact solution for problems of reasonable scale. However, the order of complexity for covering problems that we are interested in speech processing is about millions of sentences by thousands of units. In this framework, (Chevelu et al., 2007) adapts the heuristics introduced by (Caprara et al., 1999) to take into account the constraint of multi-representation : a given minimal number of instances can be required in the covering for each attribute. The obtained algorithm, called *LamSCP -Lagrangian based Algorithm for Multi-represented SCP-* is applied to extract coverings

of diphonemes, under the condition of mono-representation and 5-representation, from a large French text corpus. The results are better with *LamSCP* than the solutions found by *ASA*, offering a reduction in the cover size from 5 to 10 percents. Furthermore, *LamSCP* provides a lower bound to the *SCP* and enables to assess the real quality of the proposed solutions.

In this paper, after discussing the main steps of *LamSCP* and the lagrangian relaxation properties on which it is based, we keep comparing *LamSCP* and *ASA* to achieve a multi-represented diphoneme covering from an English text corpus. We also test these two strategies in the very constrained problem to cover triphonemes, from a English corpus and a French one, in order to study its ability to solve very large scale *SCP* instances. At last, we assert the stability of both algorithms by calculating the standard deviations of the results and the associated confidence intervals.

We want to clarify that we do not deal with the problem of choosing the right covering features. Naturally, the methods presented here can handle problems where it is necessary to compose with different kinds of features as: phonetic units, prosodic and phonological features or other speech-related features. Moreover, the notion of cost for a sentence corresponds here to its length in phone number. This figure can be modulated by other criteria. We only need that these criteria are computable. We focus in this paper on the comparison of two algorithms: a greedy based approach and a lagrangrian based one, for solving a set covering problem in the context of speech precessing. This speech processing domain brings particular event distributions known to be heavy tailed.

2. A lagrangian based method for SCP

Before introducing the *LamSCP*, we briefly review the lagrangian relaxation properties on which this algorithm is based.

2.1. Notations and principles

Let us consider a corpus $\mathcal{A}$ of n sentences composed of m distinct attributes $u_1, \ldots, u_m$ - phonological units, acoustic unit classes, prosodic attributes, etc. $\mathcal{A}$ can be represented by a matrix $A = (a_{ij})$, where a_{ij} is the instance number of u_i in the sentence s_j. We denote the unit set $\mathcal{U} = \{u_1, \ldots, u_m\}$ and we define $M = \{1, \ldots, m\}$ and $N = \{1, \ldots, n\}$. With every sentence s_j, a cost c_j is combined.

A covering of $\mathcal{U}$ is a subset of $\mathcal{A}$ which contains, for every u_i, a minimal number b_i of instances. It is described by a column vector $X = (x_j)_{j \in N}$, where $x_j = 1$ if the sentence s_j belongs to $\mathcal{U}$ and 0 otherwise. In other words, a covering is a solution $X \in \{0,1\}^n$ of the following system :

$$\forall i \in M, \sum_{j \in N} a_{ij} x_j \geq b_i \,. \tag{1}$$

If it is quite easy to determine such a covering, we want a covering with the lowest possible cost. The cost of a covering corresponds to the sum of the costs of all its elements. This *SCP* can be written as :

$$X^* = \arg \min_{\substack{X \in \{0,1\}^n \\ AX \geq B}} CX \tag{2}$$

where $C = (c_1, \ldots, c_n)$ and $B = (b_1, \ldots, b_m)^T$.

Let a column vector $\Lambda \in \mathbb{R}_+^m$, we introduce the dual lagrangian function associated with (2):

$$L(\Lambda) = \min_{X \in \{0,1\}^n} \Lambda^T B + C(\Lambda) X \tag{3}$$

where the j-th coordinate of $C(\Lambda) = C - \Lambda^T A$ is called the lagrangian cost $c_j(\Lambda)$ of s_j. The coordinates of $\Lambda = (\lambda_i)_{i \in M}$, called lagrangian multipliers, are non-negative real values and can be interpreted as a weighting of the constraints (1).

The function $L(\Lambda)$ provides a lower bound of the minimal covering cost, which permits to asset the quality of a covering. Its calculus is simple, a solution $X(\Lambda)$ of this optimisation problem in (3) is $x_j(\Lambda) = 1$ if $c_j(\Lambda) < 0$, $x_j(\Lambda) = 0$ if $c_j(\Lambda) > 0$ and $x_j(\Lambda) = 0$ otherwise. Let us notice that this lower bound is not necessary reachable by a covering cost. Moreover, $L(\Lambda)$ gives us relevant information about the usefulness of each sentence within the optimal covering. Indeed, for a given Λ and an upper bound UB of the optimal covering cost, we can compute a gap $g(\Lambda) = UB - L(\Lambda)$ which measures the quality of the relaxation. If $c_j(\Lambda) > g(\Lambda)$, we can check that any feasible solution of the *SCP* containing the sentence s_j has a cost value strictly greater than UB. Hence, s_j does not belong to the optimal solution and x_j can be fixed at zero. Similarly, if $c_j(\Lambda) < -g(\Lambda)$, s_j belongs the optimal covering and one can fix x_j to 1. Therefore, an optimal covering is made up of sentences with a low lagrangian cost (Caprara et al., 2000).

In order to obtain the best lower bound $L(\Lambda^*)$, we consider the dual problem of (2) which consists in maximizing the function L. This optimization problem is simpler than (2) : the search space is $\mathbb{R}_+^m$ on which L is continuous, concave and piecewise affine. The iterative subgradient algorithm provides a near-optimal vector Λ^* by generating a sequence $(\Lambda^k)_k$ of which the convergence fastness depends on $(g(\Lambda^k))_k$.

2.2. An introduction to LamSCP

In this paragraph, we describe the main steps and heuristics of *LamSCP* which are represented in figure 1. For more details, please refer to (Chevelu et al., 2007) and the associated references.

The algorithm is structured into three main phases, which compose the procedure called *3-phases*, as follows :

- The first phase, called *subgradient phase*, approximates Λ^*. It needs the knowledge of an upper bound of the optimal covering cost. A naive initialization of UB may be the cost of the overall corpus $\mathcal{A}$, but it is not relevant. Therefore, this initialization is carried out by calculating a first solution to the *SCP* using a greedy strategy where the score of the sentence s_j, denoted by $score_{greedy}(s_j)$, corresponds to the number of its units that are missing in the ongoing covering divided by its cost c_j.

- In the *heuristic* phase, the neighbourhood of Λ^* is explored a great number of times (usually 250). A

greedy type procedure is associated to each neighbouring vector Λ, in order to obtain a covering through the use of the lagrangian costs. More precisely, the used score function $score_{LamSCP}(s_j)$ corresponds to the number of units of the sentence s_j that are missing to the ongoing covering divided by its langrangian cost $c_j(\Lambda)$.

- From the best obtained solution, "promising" sentences are selected during the *column fixing* phase.

The residual set covering sub-problem is then processed similarly. The iteration of the *3-phases* procedure is stopped when the residual sub-problem is empty or the associated lagrangian function is too costly. More precisely, since the lagrangian function indicates a minimal cost for covering the sub-problem, its addition to the costs of the sentences already retained gives a minoration of the total cost of the solution under construction, which should not rise beyond the cost UB of the best known solution in order to be potentially more advantageous.

Since *LamSCP* aims to solve large scale *SCP*, it uses numerous heuristics to reduce the computing complexity. The most frequently heuristic consists in downsizing the problem by considering mainly the sentences with the lowest lagrangian costs. The corresponding procedures are represented by the ellipses in figure 1 :

- The *pricing* procedure, called during the subgradient phase, consists in getting the three phases to work on a subset containing sentences with a low lagrangian cost. Some other sentences are added to this subset in order to make sure that the size of the sub-corpus is sufficient with respect to the number of units to cover.

- The reduction of the problem in the procedure known as *greedy* consists in selecting the sentence within a limited subset of sentences of lowest lagrangian cost. These costs are then updated. If the maximum in this subset is bigger than the minimal lagrangian cost of the sentences that were initially excluded, the algorithm also updates the working subset.

- The *column fixing* phase and the *refining* procedure consist in really reducing the size of the problem by fixing a set of sentences and readapting the matrix as well as the constraints. The selected sentences remain selected for the whole *3-phases* procedure. They are chosen among the sentences covering rare units, or with a very low lagrangian cost.

- Finally, every time the *refining* procedure is called, the set of selected sentences is rebuilt. That step selects, up to a certain percentage of covering, the sentences that contribute the least to the gap $g(\Lambda)$.

All along the algorithm, as soon as a covering better than the current best one is derived, the upper bound UB is updated in order to improve the relaxation quality $g(\Lambda^*)$.

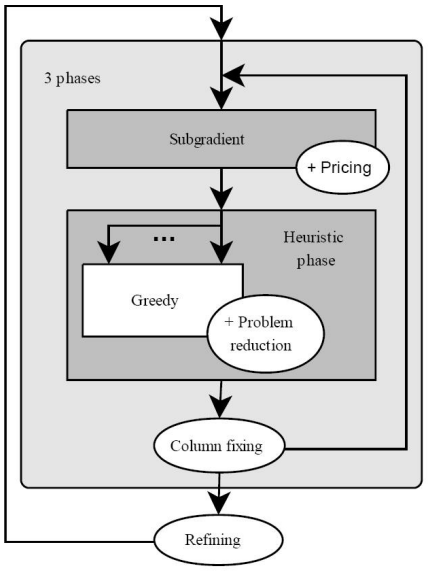

Figure 1: *The LamSCP structure. The rectangular boxes represent the steps that aim to improve the quality of the solution and the ellipses represent the step that are intended to downsize the problem.*

	Le-monde	*Gutenberg*
Corpus size (phones)	16,496,441	1,539,735
Number of sentences	172,168	53,996
Number of phonemes	35	57
Number of diphonemes	1,172	1,955
Number of triphonemes	26,443	27,477

Table 1: *Statistics of the studied corpora.*

3. Methodology

We compare *LamSCP* and *ASA* with respect to two aspects : the cost and the stability of their proposed coverings. For this purpose, large phonetically annotated text corpora are used : one in English, the other in French. The corpus in English, *Gutenberg* contains 53,996 sentences, selected by (Kominek and Black, 2003) among those of the *Gutenberg* project (Hart, 2003). The corpus in French, *Le-Monde*, is a larger corpus and counts 172,168 sentences extracted from the daily newspaper "Le Monde" (year 1997). We present in table 1 the statistics concerning both corpora.

For each corpus, existing phonemes, diphonemes and triphonemes (according the experiment and the desired attributes) were collected in order to define the set of units $\mathcal{U} = \{u_1, \ldots, u_m\}$ to be covered and their occurrences in each sentence. This information is represented by the matrix $A = (a_{ij})$ introduced in section 2.1. A covering containing at least k instances of every phoneme, diphoneme to the n-phoneme is called a k-covering of the n-phonemes. Since the main purpose is the design of textual corpora that minimize the speech duration after recording, the cost c_j of a sentence s_j corresponds to its number of phone occurrences.

In a first experiment, namely A, we carry out a 5-covering of the diphonemes of the *Gutenberg* corpus and we com-

pare the performances of *LamSCP* and *ASA* in terms of solution size, i.e. the number of phone occurrences in a solution.

In a second experiment, namely B, we realize a 1-covering of the triphonemes on the *Gutenberg* corpus, which consists in covering 29,489 units from the 53,996 available sentences. The aim is to validate the algorithms when working with strong constraints. These algorithms must be able to cope with a problem of triphonemes covering, *i.e.* with a matrix of more than 25,000 columns. To test the behavior of both algorithms when the number of sentences, and potentially the search space for a solution, are larger, we proceed the same experiment on the *Le-Monde* corpus, *i.e.* 172,168 sentences for 27,650 units.

In a last experiment, namely C, we study the stability of the results produced by both algorithms. Indeed, one of the difficulties of the greedy methodology is that the sentence score function $score_{greedy}(s_j)$ has discrete values and several sentences can have the same score. In our implementation, the greedy algorithm chooses the first coming sentence out of those that have the best current score. We would like to measure the influence of this *random* choice on the stability of the results. *LamSCP* uses heuristics that make a pre-selection from among sentences according to their lagrangian costs $c_j(\Lambda)$. Let us notice that $\Lambda \rightarrow c_j(\Lambda)$ is a continuous real-value function. In the *heuristic* phase, *LamSCP* uses greedy strategies based on the score function $score_{LamSCP}(s_j, \Lambda)$, derived from the lagrangian costs, which may be therefore more discriminant than the function $score_{greedy}(s_j)$. The obtained covering depends on the order of the sentences in the matrix, *i.e.* their column index in the matrix A. A simple solution to evaluate the stability consists in proceeding an important amount of experiments on the same *SCP* instance by randomly permutating the columns of the initial matrix A, at the beginning of each experiment.

Considering the computation time, we choose to carry out at least 40 times a 1-covering of the diphonemes on the *Gutenberg* corpus. This problem is still difficult without requiring so much computation time.

4. Results and discussion

Table 2 shows results for experiments A and B evaluating the overall quality of the covering solutions.

For each experiment, *ASA* reduces drastically the initial corpus size, in terms of number of phones and sentences, between 84 and 96%. However, for each experiment, *LamSCP* produces cheaper solutions, from 3.9 to 4.5% compared to *ASA*.

As to the mean of selected sentence lengths obtained with *LamSCP* is always higher than the *ASA* one, it may confirm that *LamSCP* makes less local choices than *ASA* especially thanks to the lagrangian vector.

Let us remind that a substantial improvement of *LamSCP* is the computation of a lower bound for the optimal covering cost, based on lagrangian relaxation principles. This lower bound is not necessarily related with a covering and may be not reachable. This information enables to establish that the optimal covering is at the maximum $1 - \frac{58,434}{61,344} = 4.7\%$ cheaper than the one found by *ASA* for experiment A, and 4.4% for experiment B (see line "Potential reduction relative to *ASA*" on table 2). Similar reasoning locates the optimal covering cost at the maximum $1 - \frac{58,434}{58,595} = 0.27\%$ smaller than the solution cost using *LamSCP* for experiment A, and respectively 0.38% and 0.55% for experiment B.

Results for experiment C are provided in table 3. As regards the average cost of a covering, the 95% confidence intervals for *LamSCP* and *ASA* are disjoint. This allows us to affirm that in terms of covering cost, solutions obtained by *LamSCP* are significantly better, around 9.7% cheaper, than the ones obtained by *ASA*. The average lower bound represents 10.46% of the average cost using *ASA*, against only 0.75% of the cost using *LamSCP*. As for stability, the relative standard deviation of this covering cost produced by *ASA* and *LamSCP* are respectively 0.46% and 0.07%. Thus, *ASA* is relatively stable, but *LamSCP* improves results by a factor 6.4. Concerning the lower bound, it hardly fluctuates with a relative standard deviation smaller than 0.05%. This seems to confirm that the lagrangian costs allow a better discrimination between sentences by providing a global information relative to the problem.

Considering the 1-covering of diphonemes (experiment C) and under comparable conditions, *LamSCP* provides a solution running in a few hundred minutes (around 225 min), whereas *ASA* needs a few minutes (around 1,5 min). Similarly, considering the 5-covering of diphonemes (experiment A), the computational time of *LamSCP* is 470 minutes whereas *ASA* needs only 22 seconds. For a triphoneme covering (experiment B), *ASA* needs 144 minutes to find a solution on the *Gutenberg* corpus compared to the 10 days of *LamSCP* (respectively 224 minutes and 11 days for the corpus *Le-Monde*). Althought the computational time of *LamSCP* seems to be huge, it is still acceptable relatively to the *SCP* size. The design of a corpus is not a frequent task and this drawback is compensated by the save of an expensive human effort for recording the reduced textual corpus, especially when several voices are needed. Inspite of the numerous heuristics in *LamSCP*, a short analysis shows that the 250 greedy procedures, which are independent, in the *heuristic* phase represent 50% of the overall execution time. It may be possible to improve the *LamSCP* computational time by using a parallel version of the *heuristic* phase.

5. Conclusion

In this paper, we compare two algorithms, *ASA* and *LamSCP*, to solve a *SCP* applied to the automatic building of linguistic corpora. The first one is based on greedy strategies and the second one on lagrangian relaxation principles. Experiments carried out in French and English to cover phonemes, diphonemes and triphonemes show that both algorithms enable to solve very large scale *SCP*. The main drawback of *LamSCP* is its computational time relatively the one of *ASA*, and its use is really relevant when the corpus size is a crucial problem for the target system. But, the lower bound, provided by *LamSCP*, permits to locate the solutions obtained by both algorithms close to the optimal covering. However, *LamSCP* gives significantly better solutions, i.e. shorter in duration. At last, both algorithms are robust to the perturbation of the matrix which repre-

Experiment A : 5-covering of diphonemes, corpus *Gutenberg*				
	Original	*ASA*	*LamSCP*	**LB**
Corpus size (phones)	1,539,735	61,344	58,595	**58,434**
Sentence number	53,996	2,289	2,099	
Sentence length mean	28.5	26.8	27.9	
"Potential" reduction relative to *ASA*			-4.5%	**-4.7%**

Experiment B : 1-covering of triphonemes				
Corpus *Gutenberg*				
	Original	*ASA*	*LamSCP*	**LB**
Corpus size (phones)	1,539,735	236,862	227,416	**226,546**
Sentence number	53,996	8,004	7,614	
Sentence length mean	28.5	29.6	29.9	
"Potential" reduction relative to *ASA*			-4.0%	**-4.4%**
Corpus *Le-Monde*				
Corpus size (phones)	16,496,441	620,568	596,422	**593,089**
Sentence number	172,168	6,991	6,436	
Sentence length mean	97.0	88.8	92.7	
"Potential" reduction relative to *ASA*			-3.9%	**-4.4%**

Table 2: The first table corresponds to a 5-covering of diphonemes of the corpus *Gutenberg*, the second one to a 1-covering of triphonemes of *Gutenberg* and *Le-Monde*. The column "Original" shows the initial corpora features. The columns "*ASA*" and "*LamSCP*" provide similar information about the covering. The column "**LB**" indicates the best lower bound found by *LamSCP*.

Experiment C : 1-covering of diphonemes, corpus *Gutenberg*			
	ASA	*LamSCP*	**LB**
Corpus size mean (phones)	14,922.0	13,460.0	**13,358.0**
Phone number std	69.3	9.7	**6.3**
Phone number relative std	0.464%	0.072%	**0.047%**
Corpus size mean (phones) 95% confidence interval	[14,899 ; 14,943]	[13,456 ; 13,463]	**[13,356 ; 13,360]**
Corpus size [minimum ; maximum]	[14,758 ; 15,052]	[13,442 ; 13,480]	**[13,344 ; 13,370]**
"Potential" reduction mean relative to *ASA*		-9.77%	**-10.46%**
Stability ratio relative to *ASA*		6.4	

Table 3: Results about algorithm stability. The table shows statistics based on 41 experiments. Each experiment corresponds to a 1-covering of diphonemes in *Gutenberg*. For each experiment, the *SCP* matrix columns are mixed. The "stability ratio" is the ratio between relative standard deviations of *LamSCP* and *ASA*.

sents the initial database, but *LamSCP* provides a sixfold improvement in stability as compared with *ASA*.

6. References

Caprara, A., M. Fischetti, and P. Toth, 1999. A heuristic method for the set covering problem. *Operations Research*, 47(5):730–743.

Caprara, A., P. Toth, and M. Fischetti, 2000. Algorithms for the set covering problem. *Annals of Operations Research*, 98(1):1–18.

Chevelu, J., N. Barbot, O. Boëffard, and A. Delhay, 2007. Lagrangian relaxation for optimal corpus design. In Wagner P., Abresh J., and Hess W. (eds.), *Proceedings of the 6th ISCA Tutorial and Research Workshop on Speech Synthesis (SSW6)*. Bonn, Germany.

François, H. 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.

François, H. and O. Boëffard, 2002. The greedy algorithm and its application to the construction of a continuous speech database. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC)*, volume 5. Las Palmas, Canary Islands, Spain.

Gauvain, J.-L., L.F. Lamel, and M. Eskenazi, 1990. Design considerations and text selection for bref, a large french readspeech corpus. In *Proceedings of the 1st International Conference of Spoken Language Processing (ICSLP)*.

Hart, M., 2003. Project gutenberg. *http://promo.net/pg*.

Kawai, H., 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 on Spoken Language Processing (ICSLP)*, volume 3. Beijing, China.

Kominek, J. and A. Black, 2003. The cmu arctic speech databases for speech synthesis research. Technical Report CMU-LTI-03-177, CMU Language Technologies Institute.

Krstulovic, S., F. Bimbot, O. Boëffard, D. Charlet, D. Fohr, and O. Mella, 2006. Optimizing the coverage of a speech database through a selection of representative speaker recordings. *Speech Communication*, 48(10):1319–1348.

Krul, A., G. Damnati, F. Yvon, Boidin C., and T. Moudenc, 2007. Adaptive database reduction for domain specific speech synthesis. In Wagner P., Abresh J., and Hess W. (eds.), *Proceedings of the 6th ISCA Tutorial and Research Workshop on Speech Synthesis (SSW6)*. Bonn, Germany.

Krul, A., G. Damnati, F. Yvon, and T. Moudenc, 2006. Corpus design based on the kullback-leibler divergence for text-to-speech synthesis application. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*. Pittsburg, USA.

Rojc, Matej and Zdravko Kacic, 2000. Design of optimal slovenian speech corpus for use in the concatenative speech synthesis system. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC)*. Athens, Greece.

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