Évaluation de méthodes de réduction de corpus linguistiques

Pierre Alain, Nelly Barbot, Olivier Boëffard, Jonathan Chevelu, Arnaud Delhay

IRISA / Université de Rennes 1 - ENSSAT
6 rue de Kerampont, B.P. 80518, F-22305 Lannion Cedex
{pierre.alain,nelly.barbot,olivier.boeffard,arnaud.delhay}@irisa.fr
jonathan.chevelu@orange-ftgroup.com

ABSTRACT

This article deals with covering methodologies in the context of automatic speech processing technologies. More precisely, we are interested in covering phonological attributes of a linguistic corpus under the constraint of a minimal duration. This goal is classically achieved by greedy algorithms which however do not guarantee the optimality of the solutions. We propose to compare the results of a new algorithm, the *LamSCP*, that calls upon the principles of lagrangian relaxation, and an agglomeration-spitting greedy algorithm to achieve an optimal covering. We conducted experiments on the Gutenberg corpus considering, phone, diphone and triphone optimal covering. The *LamSCP* provides better solutions than the greedy algorithm and enables to locate their quality by offering a lower bound to the optimization problem.
Keywords corpus design, set-covering problem, lagrangian relaxation, greedy algorithm

1. Introduction

En traitement de la parole, de nombreuses technologies font usage de méthodologies d'apprentissage automatique. Il est dans ce cas nécessaire de disposer de corpus de parole de taille et de contenu suffisants de manière à disposer de modèles efficaces. Une première optimisation de ces corpus concerne la recherche d'une diversité maximale de leur contenu, généralement en terme de diphones, diphones en contexte, triphones, marqueurs prosodiques, etc. selon l'application visée.

Afin de couvrir au mieux un ensemble d'attributs, deux stratégies sont envisageables. Une première méthode, très simple, consiste à collecter au hasard des messages linguistiques mais elle s'avère rapidement coûteuse du fait de la distribution naturelle des événements qui est de nature exponentielle (très peu d'événements sont très fréquents et beaucoup d'événements sont très rares). Cette difficulté est souvent accrue par la nécessité de disposer de plusieurs variantes d'un même événement ; c'est notamment le cas en synthèse de la parole à partir du texte. Une seconde stratégie consiste à contrôler explicitement le contenu du corpus d'apprentissage de manière à pallier les défauts de couverture des événements rares de l'approche aléatoire. Une solution consiste à sélectionner automatiquement, parmi d'immenses corpus de texte annotés linguistiquement, un sous-ensemble minimal en temps d'élocution qui couvre les attributs recherchés.

Le problème posé s'apparente à un problème de couverture d'ensemble, *SCP - Set-Covering Problem*, qui est NP-difficile. Compte-tenu de la dimension des problèmes traités, il est nécessaire de mettre en œuvre des solutions algorithmiques sous-optimales ou heuristiques. La méthodologie la plus fréquente en traitement de la parole est du type glouton par agglomération. Cet algorithme sélectionne à chaque itération une phrase de plus haut score, celui-ci reflètant sa contribution à la couverture. [5] a appliqué cette approche pour construire une base de données destinée à la reconnaissance de la parole à l'aide d'attributs de couverture organisés hiérarchiquement. [8] teste différentes variantes gloutonnes de sélection de textes, en modifiant les unités à couvrir (diphones, durée, etc.) ainsi que les "scores" des phrases selon les applications considérées. Dans [3], la méthodologie appliquée donne une priorité aux classes d'allophones les plus rares. [7] construit un corpus dont la distribution de diphonèmes/triphonèmes approche une distribution uniforme. L'algorithme glouton est dirigé par une fonction de score qui s'appuie sur la divergence de Kullback-Liebler. Dans [4], plusieurs combinaisons d'algorithmes gloutons ont été appliquées à la construction d'un corpus de synthèse de parole. D'après ces travaux, la meilleure stratégie serait l'application d'un glouton par agglomération suivi d'un glouton inversé -*cracheur*. Lors de la phase d'agglomération, le score d'une phrase correspond au nombre d'unités qu'elle contient et qui restent à couvrir, normalisé par la longueur de la phrase. Lors de la phase cracheuse, à chaque itération, la phrase redondante la plus longue en phones est supprimée de la couverture. On appellera cet algorithme *ASA*, *Agglomeration and Spitting Algorithm*.

Comme alternative à un algorithme glouton, sous-optimal par nature, [2] propose une solution basée sur la relaxation lagrangienne. En effet, la résolution du *SCP* par relaxation lagrangienne peut fournir une solution exacte pour des problèmes de taille raisonnable. Cependant, en traitement de la parole, les *SCP* étant de l'ordre de quelques millions de phrases par quelques milliers d'unités, [2] utilise des heuristiques proposées dans [1] pour résoudre des problèmes d'ordonnancement relatifs aux chemins de fer. Ces heuristiques ont été adaptées pour répondre aux contraintes de multi-représentation : un seuil minimal de représentants d'une même unité peut être exigé. L'algorithme proposé, dénommé *LamSCP -Lagrangian based Algorithm for Multi-represented SCP-* est appliqué pour

extraire des couvertures de diphones du corpus en langue française *Le Monde*. Ces résultats sont comparés à la meilleure stragégie gloutonne, *ASA*, et sont de 5 à 10 pourcents meilleurs. De plus, le *LamSCP* fournit un minorant au coût optimal de couverture et permet d'évaluer la qualité des résultats.

Dans ce papier, nous présentons succinctement le *LamSCP*. Nous évaluons le *LamSCP* et l'*ASA* sur un corpus de langue anglaise pour obtenir des couvertures mono- et multi-représentée de diphonèmes. Nous les comparons également dans le cas très contraint d'une couverture de triphonèmes. Nous ne traitons pas du choix des unités à couvrir. Les méthodes présentées ici sont adaptables à la couverture d'attributs de différente nature (phonétique, prosodique, phonologique, *etc*). Nous nous intéressons uniquement aux capacités de ces deux algorithmes à résoudre des problèmes de couverture d'ensemble dans un contexte de parole, celui-ci correspondant à des distributions naturelles d'événements bien particulières.

2. Réduction par relaxation lagrangienne

Avant de présenter le *LamSCP*, on rappelle les propriétés de la relaxation lagrangienne sur lesquelles repose cet algorithme.

2.1. Notations et principes

Soit un corpus $\mathcal{A}$ de n phrases composées de m unités possibles distinctes. $\mathcal{A}$ peut se résumer à une matrice $A = (a_{ij})$, où a_{ij} est le nombre de représentants de l'unité u_i présents dans la phrase s_j. On note l'ensemble des unités $\mathcal{U} = \{u_1, \ldots, u_m\}$, et on pose $M = \{1, \ldots, m\}$ et $N = \{1, \ldots, n\}$. À chaque phrase s_j est associé un coût c_j.

Une couverture de $\mathcal{U}$ est un sous-ensemble de $\mathcal{A}$ que l'on représente par un vecteur binaire $X = (x_j)_{j \in N}$, où $x_j = 1$ si la phrase s_j est dans la couverture et 0 sinon, et comportant pour chaque u_i un nombre minimal b_i de représentants. Une couverture est une solution $X \in \{0,1\}^n$ du système suivant :

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

S'il est simple de déterminer une couverture, on cherche ici à déterminer une couverture de coût minimal. Le coût d'une couverture correspondant à la somme des coûts de chacun de ses éléments, on peut écrire formellement ce problème d'optimisation de la façon suivante :

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

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

Soit $\Lambda \in \mathbb{R}_+^m$, on introduit la fonction lagrangienne duale associée à (2) par

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

où la j-ème coordonnée de $C(\Lambda) = C - \Lambda^T A$ désigne le coût lagrangien $c_j(\Lambda)$ associé à s_j. Les coordonnées de Λ sont appelées multiplicateurs de Lagrange et peuvent être interprétées comme une pondération de chacune des contraintes (1).

La fonction $L(\Lambda)$ présente l'intérêt de minorer le coût optimal de la couverture, permettant ainsi d'évaluer a posteriori la qualité d'une couverture. $L(\Lambda)$ est simple à calculer, une solution $X(\Lambda)$ du problème d'optimisation dans (3) étant $x_j(\Lambda) = 1$ si $c_j(\Lambda) < 0$ et $x_j(\Lambda) = 0$ sinon. Elle fournit en outre une information pertinente pour la sélection des phrases : soit UB un majorant du coût optimal de couverture et $g(\Lambda) = \mathrm{UB} - L(\Lambda)$, si $c_j(\Lambda) > g(\Lambda)$ alors s_j n'appartient pas à la couverture optimale et si $c_j(\Lambda) < -g(\Lambda)$ alors s_j appartient à la couverture optimale. Plus $g(\Lambda)$ est petit, meilleure est la qualité de la relaxation. Ainsi, une couverture optimale sera constituée de phrases ayant un faible coût lagrangien.

Pour obtenir le meilleur minorant, on recherche le maximum de la fonction L. Cette optimisation est plus facile que le problème (2) car l'espace de recherche est $\mathbb{R}_+^m$ et L est continue, concave et affine par morceaux. Une méthode itérative de type sous-gradient fournit un vecteur Λ^* sous-optimal en générant une suite $(\Lambda^k)_k$ dont la rapidité de convergence dépend de $(g(\Lambda^k))_k$.

2.2. Algorithme LamSCP

Nous présentons l'algorithme *LamSCP* dont les principales étapes sont représentées dans la figure 1. Pour plus de détails, on renvoie à [2] et ses références.

L'algorithme se structure en trois phases principales comme indiqué figure 1. La première phase, dite *sous-gradient*, approxime Λ^*. Dans la phase *heuristique*, le voisinage de Λ^* est exploré un grand nombre de fois. À chaque vecteur voisin, une procédure de type glouton (*gloutons*) est associée afin d'obtenir une couverture à l'aide des coûts lagrangiens associés. À partir de la meilleure solution obtenue, des phrases "prometteuses" sont alors fixées lors de la phase dite de *fixation de colonnes*. Le problème de couverture résiduel est alors traité de façon similaire. L'itération des trois phases est stoppée quand le problème résiduel est vide ou le minorant associé indique un coût trop élevé.

Pour diminuer la complexité calculatoire, l'heuristique la plus fréquente réduit la taille du problème en considérant principalement les phrases de plus faibles coûts lagrangiens. La procédure *élagage*, appelée au cours de la phase *sous-gradient*, consiste à appliquer les trois phases sur un sous-ensemble constitué de phrases de faible coût lagrangien auxquelles sont ajoutées d'autres phrases afin de garantir une taille suffisante du sous-corpus par rapport au nombre d'unités à couvrir. La réduction du problème dans la procédure *gloutons* consiste à sélectionner la phrase parmi un ensemble limité de phrases de plus faible coût lagrangien. Ces coûts sont ensuite mis à jour. Si leur maximum parmi ce sous-ensemble est supérieur au coût lagrangien minimal des phrases a priori exclues, l'algorithme remet également à jour le sous-ensemble de travail. Enfin la phase *fixation de colonnes* et la procédure *raffinage* réduisent réellement la taille du

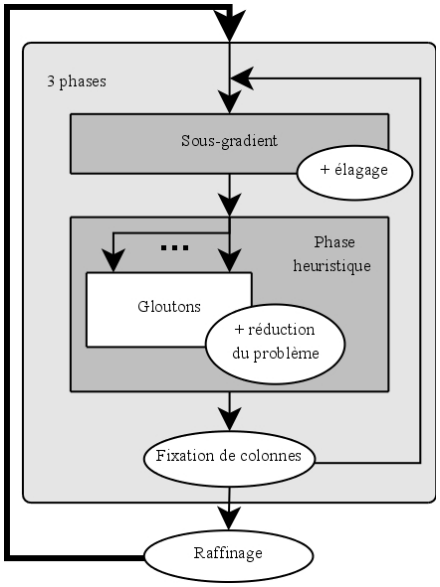

Fig. 1: *Structure du LamSCP. Les rectangles indiquent les étapes d'optimisation de la couverture et les ellipses celles de réduction de la taille du SCP.*

problème en fixant des phrases et en réadaptant la matrice et les contraintes.

Les phrases sélectionnées pendant la phase de *fixation de colonnes* le sont pour tout le reste de la procédure *3-phases*. Elles sont choisies parmi les phrases couvrant des unités rares ou à très faible coût lagrangien.

Enfin, à chaque appel de la procédure *raffinage*, l'ensemble des phrases fixées est remis en cause. Cette étape sélectionne les phrases qui contribuent le moins possible au saut de dualité, estimé par $g(\Lambda^*)$, et à hauteur d'un certain pourcentage de couverture.

Tout au long de l'algorithme, dès qu'une meilleure couverture est déterminée, la borne UB est mise à jour, afin d'améliorer la qualité de la relaxation lagrangienne indiquée par $g(\Lambda^*)$.

3. Méthodologie expérimentale

Une comparaison des algorithmes *LamSCP* et *ASA* s'effectue selon deux axes. Le premier axe porte sur le coût des couvertures calculées par chaque algorithme, que ce soit par rapport au coût minimal de couverture qu'indique le minorant fournit par le *LamSCP*, ou lors d'un passage à l'échelle. Le second axe traite du contenu des solutions obtenues.

Les expériences sont menées sur le corpus *Gutenberg* composé de textes en langue anglaise [6]. *Gutenberg* contient 53 996 phrases annotées phonétiquement et couvre 57 phonèmes (compte tenu de l'accentuation des voyelles), 1 955 diphonèmes et 27 477 triphonèmes. La taille de cette base est de 1 539 735 phones.

On appelle une "k-couverture des l-phonèmes" une couverture qui contient au moins k représentants de chaque phonème, diphonème jusqu'au "l-phonème" (i.e. triphonème si $l = 3$, diphonème si $l = 2$).

Dans les premières expériences, on compare les performances du *LamSCP* et du glouton *ASA* en terme de taille de couverture en phones. C'est donc naturellement que le coût d'une phrase correspond à sa durée en phones et le coût d'une couverture à la somme du coût des phrases qui la composent. L'expérience A porte sur une 1-couverture en diphonèmes du corpus *Gutenberg*, soit $n_c = 2\ 012$ unités à couvrir, et l'expérience B sur une 2-couverture de ces mêmes unités. Parmi ces 2012 unités, 105 d'entre elles n'apparaissent qu'une seule fois dans *Gutenberg*. L'expérience B doit donc satisfaire $n_c = 3919$ "contraintes" de couverture.

Dans une troisième expérience, notée C, on réalise une 1-couverture de triphonèmes. Cela correspond à couvrir $n_c = 29\ 489$ unités. L'objectif est de valider le fonctionnement des algorithmes lors d'un passage à l'échelle et de dimensionner la taille nécessaire pour qu'un sous-corpus contienne la totalité des triphonèmes. Nous pourrons alors comparer les contenus des corpus réduits issus des expériences A et B en terme de triphonèmes.

4. Résultats et discussion

Le tableau 1 présente les résultats des expériences A, B et C dont l'objectif est d'évaluer la qualité des solutions. Pour chaque expérience, l'*ASA* réduit très fortement le corpus initial aussi bien en nombre de phones qu'en nombre de phrases, de l'ordre de $84,6$ à $99,0\%$. Cependant, le *LamSCP* produit des solutions plus économiques pour chaque expérience, de 4 à $10,6\%$ par rapport à *ASA*. En effet, le *LamSCP* sélectionne les phrases selon leur coût lagrangien $c_j(\Lambda)$ (fonction continue) qui a un rôle plus discrimimant que le coût c_j utilisé par l'*ASA*. Le *LamSCP* effectue ainsi moins de choix locaux. On peut remarquer que plus le nombre de "contraintes" n_c est élevé, plus cet écart relatif entre les coûts des solutions obtenues par les deux algorithmes diminue. L'augmentation de n_c tend à diminuer le nombre de phrases de coût identique, et donc à restreindre le nombre de choix locaux pour les deux algorithmes ; l'impact des coûts lagrangiens est alors réduit.

De même, plus le *SCP* est contraint, plus longues sont les phrases sélectionnées dans les couvertures, aussi bien par l'*ASA* que par le *LamSCP*. La longueur moyenne des phrases choisies par le *LamSCP* est toujours supérieure à celles extraites par l'*ASA*. Cela semble confirmer les interprétations précédentes, quant au comportement des algorithmes relativement aux choix locaux.

Rappellons qu'un apport important du *LamSCP* est le calcul d'un minorant du coût optimal de couverture. À ce minorant ne correspond pas nécessairement une couverture, il n'est peut-être pas atteignable, mais il permet d'évaluer la qualité des solutions obtenues par les divers algorithmes. On peut ainsi affirmer que la couverture optimale est au maximum $1 - \frac{13\ 352}{15\ 058} = 11,3\%$ moins coûteuse que la solution fournie par l'*ASA* pour l'expérience A, respectivement $7,9\%$ pour B et $4,3\%$ pour C. Des calculs similaires situent les coûts optimaux de couverture à moins de 0.75% pour l'expérience A, respectivement 0.66% pour B et 0.38% pour C, des coûts des solutions

k-couverture de n-phonèmes	Expérience A $k=1,\ l=2$			Expérience B $k=2,\ l=2$			Expérience C $k=1,\ l=3$		
n_c	2012			3919			29477		
Algorithme	*ASA*	*LamSCP*	**(Min)**	*ASA*	*LamSCP*	**(Min)**	*ASA*	*LamSCP*	**(Min)**
Taille (phones)	15 058	13 454	**(13 352)**	27 615	25 585	**(25 414)**	236 862	227 416	**(226 546)**
Taille/n_c	7,4	6,6	**(6,6)**	7,0	6,6	**(6,4)**	8,0	7,7	**(7,6)**
Réduction/*Gutenberg*	-99,0%	-99,1%		-98,0%	-98,3%		-84,6%	-85,2 %	
Nombre de phrases	621	516		1 072	946		8 004	7 614	
Taille/phrase	24,3	26,1		25,7	27,0		29,6	29,9	
Réduction/*ASA*	0%	-10,6%	**(-11,3%)**	0%	-7,3%	**(-7,9%)**	0%	-4,0%	**(-4,3%)**

Tab. 1: Résultats de couvertures de diphonèmes et triphonèmes par *ASA* et *LamSCP*. Les colonnes **(Min)** indiquent les statistiques relatives aux minorants.

obtenues par le *LamSCP*.

	Expérience A $k=1,\ l=2$		Expérience B $k=2,\ l=2$	
Algorithme	*ASA*	*LamSCP*	*ASA*	*LamSCP*
Taille/*Gutenberg*	0,97%	0,87%	1,8%	1,6%
% de triphonèmes	23,7	22,0	33,2	31,7
% relatif de triph.	74,9	73,0	84,2	83,1

Tab. 2: Représentation des triphonèmes dans les couvertures de diphonèmes

On peut également remarquer une certaine proportionnalité entre le nombre de "contraintes" n_c et la taille des solutions, le facteur allant de 6,6 à 8,0 selon les expériences. Ainsi, si une couverture des phonèmes et diphonèmes reste raisonnable en durée d'enregistrement (de l'ordre de 14 000 phones soit 20 minutes environ), l'ajout des triphonèmes dans l'ensemble des unités à couvrir multiplie au minimun par un facteur 17 la taille de la solution. La nécessité de couvrir tous les triphonèmes est discutable selon la tâche visée et il est intéressant de considèrer le taux de couverture des triphonèmes dans les corpus réduits issus des expériences A et B. Le tableau 2 indique le pourcentage de triphonèmes (phonèmes et diphonèmes compris) couverts par ces ensembles ainsi que leur représentativité dans *Gutenberg*, compte-tenu de leurs instances. On peut observer que ces corpus réduits, en dépit de leur petite taille, couvrent plus de 1/5 des triphonèmes et de $73,0$ à $84,2\%$ des instances de triphonèmes dans *Gutenberg*. Cela se justifie par la distribution naturelle des unités (loi de Zipf).

Enfin, pour ce qui est des temps de calcul, le *LamSCP* est beaucoup plus lent que l'*ASA*. À conditions comparables, le *LamSCP* nécessite quelques centaines de minutes pour les expériences A et B et l'*ASA* à peine quelques minutes (ratio $\simeq 150$). Quant à C, l'*ASA* met 114 minutes et le *LamSCP* 11 jours 18 heures. Cette différence est somme toute relative, l'opération de construction d'un corpus n'étant pas si fréquente. La réduction d'un corpus textuel est réalisée avant la phase d'enregistrement de la base sonore. Le cas échéant, le corpus réduit peut être utilisé pour l'enregistrement de plusieurs voix. Le coût supplémentaire engendré par le *LamSCP* pour la construction d'un tel corpus est uniquement de la ressource machine. Selon la tâche visée et la taille du problème, un temps de calcul de l'ordre de plusieurs jours peut s'avérer largement acceptable.

5. Conclusion

Dans cet article, nous comparons deux algorithmes, l'*ASA* et le *LamSCP*, pour répondre à un *SCP* relatif à la construction automatique de corpus linguistiques. L'un est basé sur des stratégies gloutonnes et l'autre sur des propriétés de la relaxation lagrangienne. Les expériences menées pour couvrir les phonèmes, diphonèmes et triphonèmes montrent que ces algorithmes sont capables de résoudre des *SCP* de grande taille. Le minorant, fourni par le *LamSCP*, permet de localiser les couvertures des deux algorithmes très proches des solutions optimales. Si le *LamSCP* donne de meilleurs résultats (jusqu'à 10% par rapport à l'*ASA*), son temps de calcul est cependant beaucoup plus important. Son utilisation n'est justifiée que si la taille de corpus est un point critique dans la construction du système visé. L'*ASA* est donc une bonne méthode en terme de qualité de solution.

Références

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

[2] J. Chevelu, N. Barbot, O. Boëffard, and A. Delhay. Lagrangian relaxation for optimal corpus design. In *Proc. of the ISCA Tutorial and Research Workshop on Speech Synthesis (SSW6)*, pages 211–216, 2007.

[3] H. François and O. Boëffard. Design of an optimal continuous speech database for text-to-speech synthesis considered as a set covering problem. In *Proc. of Eurospeech*, pages 829–833, 2001.

[4] H. François and O. Boëffard. The greedy algorithm and its application to the construction of a continuous speech database. In *Proc. of LREC*, volume 5, pages 1420–1426, 2002.

[5] J.-L. Gauvain, L.F. Lamel, and M. Eskenazi. Design considerations and text selection for bref, a large french read-speech corpus. In *Proc. of ICSLP*, pages 1097–1100, 1990.

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

[7] A. Krul, G. Damnati, F. Yvon, and T. Moudenc. Corpus design based on the kullback-leibler divergence for text-to-speech synthesis application. In *Proc. of ICSLP*, pages 2030–2033, 2006.

[8] J.P.H. Van Santen and A.L. Buchsbaum. Methods for optimal text selection. In *Proc. of Eurospeech*, pages 553–556, 1997.