ATIAM PAM : AutoDj-Rapport
Etat de l'art

Maxime Arbisa, Léo Ayrthifis, Samuel Bell-Bell,Pierre Mahé,

21 janvier 2016

image ici

Table des matières

1 Introduction

2 Alignement des accords

2.1 Accord

Nous lirons les accords simplifié extrait des morceaux avec la notation simplifié sous forme de notation américaine des accords uniquement majeur ou mineur des demie-ton.

2.1.1 Matrice de similarité

Nous avons deux type de matrice de similarité : une naïve basé sur match (+10) et mismatch (-10) l'autre matrice de similarité est basé sur la distance des échelles chromatique entre deux accord

2.2 Algorithme de Needleman-Wunsch

2.2.1 Presentation de l'algorithme

Cette algorithme a pour but de présenté l'alignement maximal global entre deux chaîne de caractère, il a été devellopé pour trouver les similarité entre les acides aminé de deux protéine. Nous réutilisons ce principe pour calculé l'alignement global maximal entre les accords de deux morceaux.
Cette algorithme est programmer sous forme de programmation dynamique il assure donc l'optimalité de la solution et est beaucoup plus rapide qu'un algorithme naïf car la solution dépend des sous solutions précédente.
Il existe 3 opérations possible pour construire de la matrice de coût entre les deux chaîne de caractere :

- match : les deux lettres courante sont les mêmes ;
- mismatch : les deux lettres courantes ne sont pas les même ;
- Indel (INsertion ou DELetion) : une lettre est aligné avec un gap dans l'autre chaine de caractére

Pour remplir la matrice on remplit la premiere colonne est ligne avec le poids d'un indel * positionColonne ou indel * positionLigne en fonction. ensuite on se base sur le principe d'optimalité des opérations possible pour remplir le reste de la matrice de coût M et S la matrice de similarité

$$M_{i,j} = max(M_{i-1,j-1} + S(A_i, B_j), M_{i,j-1} + d, M_{i-1,j} + d)$$

Backtracking : Tout en construisant la matrice de coût , on construire une matrice d'antécédent pour savoir d'où l'on viens par rapport à la case courante avec les mouvements réprésentant match (diagonal), mismatch(gauche) et Indel (monter)

2.3 Smith-Waterman

2.3.1 Presentation de l'algorithme

A la différence de Needleman-Wunsch , Smith-Warerman a pour but de présenté l'alignement maximal local entre deux chaîne de caractère. La matrice de coût Needleman-Wunsch est non-negative d'où la régle ajouter par rapport a Needleman-Wunsch de mettre a 0 si le score de l'élément suivant deviens négatif. La premiere ligne et la premiere colonne est initialisé a 0.

$$M_{i,j} = max(M_{i-1,j-1} + S(A_i, B_j), M_{i,j-1} + d, M_{i-1,j} + d, 0)$$

Backtracking On cherche la position où se trouve le ou les maximums de la matrice de coût puis en suivant la matrice d'antécédent à partir de la position trouvé , on parcour jusqu'a la rencontre d'un élément de la matrice de coût à zeros. On récupère alors cette chaine.

2.4 Affine GAP Cost

Nous avons commencé par un linear cost, affine gap coast , permet de construire deux sequence pas parfaitement similaire mais sans trou avec un indel au milieu.
en effet l' affine gap cost est basé sur le séparation du gap cost en 2 : open gap et ext gap.
Cette méthode complexifi le backtracking de smith-waterman et Needleman-Wunsch qui demande alors l'existance de trois matrice de traceback.

3 Transformer Q-constante (CQT)

La transformer Q-constante elle a pour altérer au problème de résolution de la transformé de Fourier on utilise la Q-constante-transformation qui change la résolution fréquentielle en fonction des fréquence considérées.

$$Q = \frac{f_k}{f_{k+1} - f_k}$$

$$Q = \frac{f_k}{B_w} = \frac{f_k}{C_w/L} = \frac{f_k * L}{C_w}$$

En partant de la TFCT mais on la fenetre vari pour obtenir un facteur de qualité constante. Ce facteur est basé sur des propriétés de perception de l'oreille humaine.

$$X(n,k) = \sum_{n=0}^{N-1} x(n)W(n-m)e^{-\frac{2j\pi mk}{N}}$$

3.1 CQT classique

L'algorithme pour créé une CQT classique se déroule en trois étape : Dans un premier temps on calcule me kernel de la Q-transforme, où un élément du kernel est le produit entre la fenetre de la q-constante actuel et de l'exponentiel.

$$X(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)W(n-m)e^{-\frac{2j\pi mk}{N}} \;\; kern(n) = W(n-m)e^{-\frac{2j\pi mk}{N}}$$

avec

$$N = N[k] = Q\frac{f_s}{f_k}$$

Ensuite on fenetre le Kernel avec une fenetre de Hann Enfin on multiplie le signal par le kernel $\sum_{n=0}^{Nb_k} x[t] * kern(k)$ La complexité temporel de cette version s'avére très grande , on multiplie a chaque fois les frames de la TFCT par un kernel qui a de nombreux élément, de plus Q pour être précis doit avoir un nombre de période plutôt grand , le nombre de calcul est donc de Q*nbElemKern*nbframe de plus l'hopsize doit faire la taille de la plus petit element du kernel sur trois (puisqu'une fenêtre fait 75% de recouvrement

3.2 CQT efficiente

Une deuxième méthode appellé Q-transfort-efficiant , les deux premiere étape sont semblable à la précédent ensuite on applique une TFCT au signal x et on utilise le produit spectral entre la $KERN = TF(kern)$ puis puisqu'une convolution en temporel est un produit en fréquence on calcul $X(k,n) * KERN(f)$ cette méthode permet de gagné une grande complexité temporel par rapport a la méthode précédente.

4 Tracking de beat

Pour suivre l'évolution du bpm d'un morceau au cour du temps nous avons utilisé le schémas suivant :

Les onsets représente l'attaque du son, pour les détecter on a besoin de la TFCT du signal avec des une fenetre de 46ms et un overlap de 10ms (ce qui permet de construire une TFTC à une échelle plutôt proche de la perception humaine)

$$X(n,k) = \sum_{m=-N/2}^{N/2-1} x(hn+m)w(m)e^{-\frac{2j\pi mk}{N}}$$

Il existe plusieurs fonction de detection de onset, nous avons utilisé le Spectral Flux (SF) qui se calcul de la manière suivante : on calcule la différentiation trame par trame suivie d'une half-wave rectification puis on l'a somme

$$SF(n) = \sum_{m=-N/2}^{N/2-1} H(\mid X(n,k)\mid - \mid X(n-1,k)\mid)$$

Une fois que nous avons notre fonction d'onset on peut alors construire notre rythmogram pour cela on fait une fonction d'autocorrelation du Spectral Flux sur des trames de 8 secondes. Ensuite on concaténe alors chaque vision temporel voir période

Avec une FFT à la place d'une autocorrelation on se retrouve dans le domaine fréquenciel le bpm étant une fréquence, mises a l'échelle on obtient alors les bandes réprésentant les bpm.

Une fois le rythmogram du domaine fréquenciel construit, nous répérons le bpm pour chaque trame. Nous souhaitons alors trouvé les positions battue réel dans le morceau en sample, pour cela il suffit de convolué un peigne de dirac, on retrouve alors le decalage en recupérant la distance entre le onset maximal de la trame et l'élément du peigne de dirac le plus proche. On réapplique sur l'ensemble de frame cette méthode est on retient les positions trouvé

5 Detune

Le detunage est la différence entre une fréquence et la fréquence de note la plus proche, une guitare mal accordé est détuné par exemple.
Pour ce faire nous avons crée un coefficient de détunage, le calcul est simple. Nous trouvons sur plusieurs petit extrait du même morceau via le Produit Spectral la fréquence qui ressort entre 200 et 900 hz afin d'évité les kicks notament.cette fréquence trouvé est comparé
à la note midi en fréquence la plus proche qui sera appellé f_{ref} on a donc $coef = f_{trouve}/f_{ref}$ puis on moyenne chacun de ces coefficien sur le nombre d'extrait choisis pour avoir un résulta plus stable et réaliste.