Une Approche Lagrangienne Appliquée aux Problèmes d'Alignement dans la Bioinformatique

S. Balev

Université du Havre, LITIS EA 4108
BP 540, 76058 Le Havre - FRANCE
stefan.balev@univ-lehavre.fr

Mots-clefs : programmation en variables binaires, relaxation lagrangienne, bioinformatique

1 Introduction

Une grande partie des problèmes bioinformatiques sont des problèmes d'alignement. Les problèmes d'alignement de deux séquences sont faciles. Ils sont traités par des algorithmes de programmation dynamique. Mais les problèmes d'alignement d'une séquence et une structure ou de deux structures sont NP-durs et nécessitent des algorithmes spécialises de résolution exacte ou approchée.

Ici nous nous intéressons à un problème particulier d'alignement séquence-structure. La résolution de ce problème fait partie d'une classe de méthodes de prédiction de la structure des protéines à partir de leur séquence d'acides aminés. Nous présenterons une approche basée sur une formulation en variables binaires et une relaxation lagrangienne. Cette approche peut être appliquée sur d'autres problèmes d'alignement séquence-structure ou structure-structure.

2 Formulation du problème

Une structure est une description abstraite d'une famille structurelle de protéines. C'est un ensemble ordonné de m blocs. Le bloc i a une longueur fixe l_i et représente une séquence contiguë d'acides aminés. Les blocs correspondent aux éléments conservés de la structure secondaire (hélices α et feuillets β). La structure contient également une description des interactions entre des paires d'acides aminés appartenant aux blocs. Ces informations sont regroupées dans un graphe de contact avec des sommets représentant les blocs et un ensemble d'arêtes E représentant des interactions entre eux.

Un alignement entre une séquence requête de longueur N acides aminés et une structure est une couverture de zones de la requête par les blocs de la structure. Les blocs doivent préserver leur ordre et ne peuvent pas se chevaucher (cf Fig. 6). Un alignement est complètement déterminé par les positions de début des blocs. Pour simplifier nos notations, nous utiliserons des positions relatives. Si la position absolue du bloc i est j, sa position relative est $j - \sum_{k=1}^{i-1} l_k$. Ainsi, les positions possibles de chaque bloc sont entre 1 et $n = N+1-\sum_{i=1}^{m} l_i$ et l'ensemble d'alignements possibles est $T = \{\pi = (\pi_1, \ldots, \pi_m) \ : \ 1 \leq \pi_1 \leq \ldots \pi_m \leq n\}$. Il est facile de démontrer que le nombre d'alignements possibles est $|T| = \binom{m+n-1}{m}$, ce qui est un nombre énorme même pour des instances de petite taille.

Le score d'un alignement est défini par un ensemble de coefficients c_{ijkl}, $(i, k) \in E$, $1 \leq j \leq l \leq n$, où c_{ijkl} est le score généré par l'interaction entre les blocs i et k quand le bloc i est placé à la position j et le bloc k occupe la position l. Ainsi, le but est de trouver l'alignement de score minimum parmi tous les alignements de T.

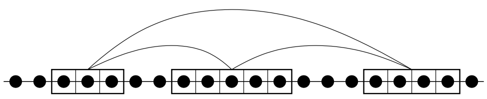

FIG. 6 – Exemple d'alignement d'une séquence requête de longueur 20 et une structure contenant 3 blocs de longueurs 3, 5 et 4

Soient y_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$ des variables binaires, telles que $y_{ij} = 1$ ssi le bloc i occupe la position j. Soit Y le polytope défini par les contraintes suivantes :

$$\sum_{j=1}^{n} y_{ij} = 1 \qquad i = 1, \ldots, m \tag{1}$$

$$\sum_{l=1}^{j} y_{il} - \sum_{l=1}^{j} y_{i+1,l} \geq 0 \qquad i = 1, \ldots, m-1,\ j = 1, \ldots, n-1 \tag{2}$$

$$y_{ij} \geq 0 \qquad i = 1, \ldots, m,\ j = 1, \ldots, n \tag{3}$$

Ainsi, $Y \cap B^{mn}$ décrit exactement l'ensemble des alignements possibles. Il est intéressant de noter que tous les sommets du polytope Y sont entiers [5] et donc on peut optimiser toute fonction linéaire des variables y sans imposer une contrainte d'intégralité de celles-ci.

Malheureusement, notre fonction de score ne peut pas être exprimée comme une fonction linéaire des variables y. En fait, il s'agit d'une fonction quadratique et non-convexe $f(y) = \sum c_{ijkl} y_{ij} y_{kl}$. Afin de linéariser cette fonction, nous introduisons des variables binaires $z_{ijkl} = y_{ij} y_{kl}$. Les variables z sont reliées aux variables y par les contraintes suivantes :

$$y_{ij} = \sum_{l=k}^{n} z_{ijkl} \qquad (i,k) \in E,\ j = 1, \ldots, n \tag{4}$$

$$y_{kl} = \sum_{j=1}^{l} z_{ijkl} \qquad (i,k) \in E,\ l = 1, \ldots, n \tag{5}$$

Avec ces contraintes supplémentaires le polytope correspondant n'est plus intégral et la relaxation linéaire ne donne pas la solution optimale du problème.

Différents modèles de programmation en variables binaires, plus ou moins proches de celui que nous venons de décrire ont été proposés [4, 1]. La résolution de ces modèles par la méthode de branch-and-bound standard utilisant des relaxations linéaires est relativement efficace. Typiquement l'arbre de recherche est très réduit, souvent ne contenant qu'un seul noeud. Mais pour des instances de grande taille le nombre de variables et de contraintes est tellement grand que même la relaxation linéaire devient très difficile à résoudre. C'est la raison pour laquelle nous utiliserons une relaxation lagrangienne qui est à la fois plus facile à résoudre et fournit des meilleures bornes.

3 Relaxation lagrangienne

Comme nous l'avons vu, les contraintes « difficiles » et donc candidats de relaxation sont (4) et (5). Si l'on relaxe toutes ces contraintes (cf Fig. 7b), la relaxation devient très facile à résoudre. Pour chaque interaction $(i, k) \in E$, il suffit de choisir des positions j et l qui minimisent le coût associé à z_{ijkl} et indépendamment de choisir les meilleurs positions de blocs par rapport

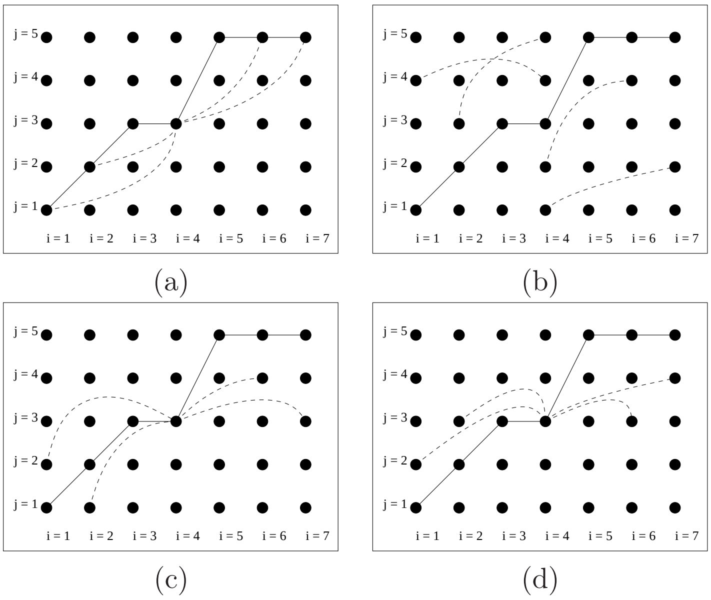

FIG. 7 – Exemple d'une instance du problème d'alignement avec $m = 7$ blocs et $n = 5$ positions libres. Les colonnes correspondent aux blocs et les lignes aux positions. L'ensemble d'interactions entre blocs est $E = \{(1,4),(2,4),(4,6),(4,7)\}$. Les lignes solides relient les points représentant les positions des blocs telles qu'elles sont définies par les variables y. Les lignes pointillées relient les points représentant les positions des blocs telles qu'elles sont définies par les variables z. (a) une solution satisfaisant toutes les contraintes. (b) les contraintes (4) et (5) sont relaxées. (c) seulement la moitié des contraintes (4) et (5) est relaxée (l'une des extrémités de chaque interaction $(i,k) \in E$). (d) comme (c), mais avec des contraintes de non-chevauchement des blocs sur les variables z.

aux variables y. La complexité d'une telle procédure est $O((m + |E|)n^2)$. On peut démontrer que dans ce cas la borne fournie par le dual lagrangien est égale à celle fournie par la relaxation linéaire.

Afin de serrer cette borne, nous ne relaxons qu'une partie des contraintes (4) et (5). Pour chaque interaction $(i,k) \in E$, nous choisissons l'une des extrémités (soit i, soit k) et nous la « détachons » en relaxant soit (4), soit (5) (cf Fig. 7c). Cette relaxation n'est pas plus difficile à résoudre que la précédente. En effet, considérons un bloc i placé à la position j. Par une simple procédure de programmation dynamique on peut trouver les positions optimales de toutes les blocs k, tels que (i,k) ou (k,i) est une interaction avec extrémité détachée k. En plus on peut imposer des contraintes de non-chevauchement sur ces blocs, comme le montre Fig. 7d. En prenant en compte ces contributions de score pour chaque point (i,j), il est facile de trouver les positions optimales des blocs par rapport aux variables y. Pour une explication plus détaillée de l'algorithme de résolution de la relaxation lagrangienne, voir [2]. Cette relaxation est nettement meilleure que la précédente. En plus, la complexité de résolution reste la même, $O((m+|E|)n^2)$. Dans le pire des cas $|E|$ est d'ordre m^2, mais pour les problèmes pratiques $|E|$ est d'ordre m, ce qui donne une complexité $O(mn^2)$.

Nous résolvons le dual lagrangien par une méthode d'optimisation sous-gradient standard [3] limitée à 500 itérations. À chaque itération t nous résolvons la relaxation lagrangienne avec les valeurs courantes λ^t des multiplicateurs lagrangiens. Pour l'itération suivante, $\lambda^{t+1} = \lambda^t + \theta_t s^t$, où s^t est le sous-gradient de la fonction objective pour $\lambda = \lambda^t$ et θ_t est le pas qui diminue à chaque itération.

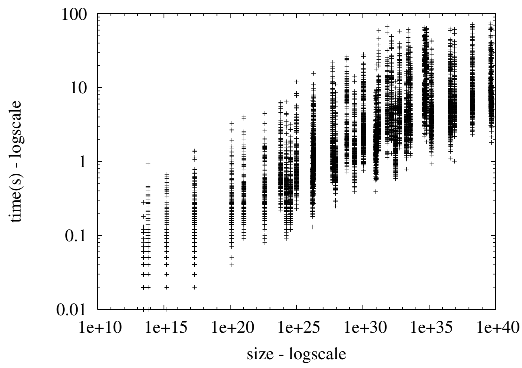

Fig. 8 – Temps d'exécution pour 9 136 instances du problème d'alignement en fonction de la taille de l'espace de recherche.

4 Résultats expérimentaux

Nous avons implémente un algorithme branch-and-bound qui utilise les bornes fournies par le dual lagrangien et nous l'avons testé sur des problèmes d'alignement de taille differente. Fig. 8 montre le temps d'exécution de 9 136 alignements en fonction de la taille de l'espace de recherche $|T|$.

La solution optimale a été trouvée en moins d'une minute pour toutes les instances sauf 34. Pour 32 de celles dernières l'optimum a été trouvé en moins de 4 minutes et seulement pour deux instances l'optimum n'a pas été trouvé au bout d'une heure. Cependant, pour ces deux instances notre algorithme a fourni en une minute une solution sous-optimale avec un gap prouvé de la valeur objective de moins de 0,1%. Il est intéressant de noter que pour 79% des instances la solution optimale a été trouvée dans la racine de l'arbre de branch-and-bound. Autrement dit, la solution produite par le dual lagrangien est réalisable pour le problème original. On observe le même phénomène quand on utilise une relaxation linéaire. L'avantage de notre méthode est que le dual lagrangien est résolu beaucoup plus rapidement par notre algorithme dédié que la relaxation linéaire par la méthode de simplexe. En titre de comparaison, le temps moyen de résolution de la relaxation linéaire par CPLEX pour une instance de taille d'ordre 10^{38} est plus d'une heure, alors que le dual lagrangien est résolu en environ 15 secondes pour des instances de la même taille.

Références

[1] R. Andonov, S. Balev, and N. Yanev. Protein threading : From mathematical models to parallel implementations. *INFORMS Journal on Computing*, 16(4) :393–405, 2004.

[2] S. Balev. Solving the protein threading problem by lagrangian relaxation. In *Proceedings of WABI 2004 : 4th Workshop on Algorithms in Bioinformatics*, volume 3240 of *LNCS/LNBI*, pages 182–193. Springer-Verlag, 2004.

[3] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1988.

[4] J. Xu, M. Li, G. Lin, D. Kim, and Y. Xu. RAPTOR : optimal protein threading by linear programming. *Journal of Bioinformatics and Computational Biology*, 1(1) :95–118, 2003.

[5] N. Yanev, R. Andonov, P. Veber, and S. Balev. Lagrangian approaches for a class of matching problems in computational biology. *Computers and Mathematics with Applications*, 55 :1054–1067, 2008.