CS 466 – Introduction to Bioinformatics – Lecture 7

Mohammed El-Kebir

September 26, 2018

Document history:

- 9/24/2018: Initial version.
- 9/26/2018: Simplified ILP.
- 9/26/2018: Typos in Section 1.2.1.

Contents

1 Protein Structure Alignment

1.1 Background

A protein achieves its biological function through its 3-D structure. The problem of predicting protein structure from an amino acid sequence is a more challenging problem than the prediction of RNA secondary structure that we saw in the last lecture. While RNA secondary structure is determined by a set of complementary base pairs, protein structure is more involved due to a larger number of residues with different chemical properties and interactions. Protein structures are typically determined using experimental techniques such as X-ray crystallography and NMR spectroscopy.

In this lecture, we consider the problem of comparing two protein structures. The key idea is that biological function is evolutionary conserved. Since function is achieved directly through structure, we expect to uncover more evolutionary relationships through pairwise protein *structure* comparison rather than pairwise protein *sequence* comparison.

These notes are based on material from [1, 2].

1.2 Problem Statement

Let Σ be the alphabet of amino acids (i.e. $|\Sigma| = 20$). Let $A \in \Sigma^m$ and $B \in \Sigma^n$ be two protein sequences. In addition, we are given the 3-D spatial coordinates of each amino acid, or *residue*, of A and B. We denote the distance between two residues i, j of A by $d_A(i, j)$. Given a distance threshold τ, we binarize the distances, yielding an $m \times m$ binary matrix $C^A = [c^A_{i,j}]$ where $c^A_{i,j} = 1$ if and only if $d_A(i, j) \leq \tau$. We call matrix C^A a contact map, whose 1-entries indicate pairs of residues that are in close contact.

An *alignment* S of A and B can be viewed as a set of non-overlapping pairs $(i, k) \in [m] \times [n]$ of residues, with the additional constraint that there for all distinct aligned pairs $(i, k), (j, l) \in S$ it holds that

$$(i < j \text{ and } k < l) \text{ or } (j < i \text{ and } l < k). \tag{1}$$

A *conserved contact* in S is defined by two aligned pairs $(i, k), (j, l) \in S$ such that $c^A_{i,j} = 1$ and $c^B_{k,l}$. We consider the following problem.

Problem 1 (Contact Map Overlap (CMO)**.** *Given two contact maps C^A and C^B of protein sequences $A \in \Sigma^m$ and $B \in \Sigma^n$, find an alignment of A and B with maximum number of conserved contacts.*

1.2.1 Graph-based formulations

There are two different ways to view CMO as a graph problem. Both representations model protein structures (A, C^A) and (B, C^B) by two graphs $G_A = (V_A, E_A)$ and $G_B = (V_B, E_B)$. There is a vertex $v_i \in V_A$ ($v_k \in V_B$) for each residue $i \in [m]$ ($k \in [n]$). There is an edge $(v_i, v_j) \in E_A$ ($(v_k, v_l) \in E_B$) for each pair $i, j \in [n]$ ($k, l \in [m]$) of residues where $c^A_{i,j} = 1$ ($c^B_{k,l} = 1$).

1. Matching graph.

   The *matching graph* $G_M = (V_A \cup V_B, E_M)$ is a complete bipartite graph, i.e. $E_M = V_A \times V_B$. An alignment in G_M is a non-crossing matching S, i.e. a subset of pairwise disjoint edges that do not cross. That is, for all distinct edges $(v_i, v_k), (v_j, v_l) \in S$ it holds that

   $$(i < j \text{ and } k < l) \text{ or } (j < i \text{ and } l < k). \tag{2}$$

   What is the score of S?

2. Product graph.

   The *product graph* $G = (V_A \times V_B, E)$ has a directed edge $((v_i, v_k), (v_j, v_l)) \in E$ if and only if $i < j$, $k < l$, $c^A_{i,j} = 1$ and $c^B_{k,l} = 1$. Observe that by definition G is a directed acyclic graph (DAG). We can draw G as a grid with vertex (v_1, v_1) in the bottom left corner and vertex (v_m, v_n) in the top right corner. An alignment in G is a set $S \subseteq V$ of vertices such that for all two distinct vertices $(v_i, v_k), (v_j, v_l) \in S$ it holds that

   $$(i < j \text{ and } k < l) \text{ or } (j < i \text{ and } l < k). \tag{3}$$

   What is the score of S?

1.3 Complexity

Recall the Clique problem, where given a simple graph $G = (V, E)$ and an integer $k \in \mathbb{N}$ we want to decide whether G has a clique of size k, i.e. a subset $V' \subseteq V$ of vertices such that $|V'| = k$ and for all distinct $v, w \in V'$ it holds that $(v, w) \in E$. The Clique problem is NP-complete. We reduce Clique to CMO, by simply setting $G_1 = G$ and $G_2 = K_k$, i.e. G_2 is the complete graph of k vertices. Clearly, this reduction takes polynomial time. Now, observe that G has a clique of size k if and only if the maximum number of conserved contacts in G_1 and G_2 is k.

1.4 Integer Linear Programming Formulation

We start with the matching graph representation $G = (V_1 \cup V_2, E_M)$. For each $(v_i, v_k) \in E_M$, we introduce a binary variable $x_{i,k} \in \{0, 1\}$ where $x_{i,k} = 1$ if and only if residue i of A is aligned with residue k of B. For each pair $(v_i, v_k), (v_j, v_l) \in E_M$ of edges, we introduce the constant $c_{i,k,j,l}$ defined as

$$c_{i,k,j,l} = \begin{cases} 1, & \text{if } c^A_{i,j} = c^B_{k,l} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Let X be the set of all alignments. This leads to the following mathematical program.

$$\max_{\mathbf{x}} \quad \sum_{i=1}^{m} \sum_{k=1}^{n} \sum_{j=i+1}^{m} \sum_{l=k+1}^{n} c_{i,k,j,l} x_{i,k} x_{j,l} \tag{5}$$

$$\text{s.t.} \quad \mathbf{x} \in X \tag{6}$$

We define X using linear constraints. To this end, we define a *pairwise incompatible set* as a subset $I \subseteq E_M$ such that for all distinct pairs $(i, k), (j, l) \in I$ it does *not* hold that ($i < j$ and $k < l$) or ($j < i$ and $l < k$). Subset I is maximal if it cannot be extended by an additional edge in E_M yielding another pairwise incompatible set I'. Equivalently, a maximal pairwise incompatible set I is a decreasing path $(i_1, k_1), (i_2, k_2), \ldots (i_m, k_m)$ where $i_1 \geq i_2 \geq \ldots i_m$ and $k_1 \leq k_2 \leq \ldots k_m$ in the product graph G (assuming $m \geq n$). Let $\mathcal{I}$ be the set of all maximal pairwise incompatible pairs. *Exercise:* How large is $\mathcal{I}$?

We replace (6) by the following constraints.

$$\sum_{(i,k) \in I} x_{i,k} \leq 1 \qquad \forall I \in \mathcal{I} \tag{7}$$

$$x_{i,j} \in \{0, 1\} \qquad \forall i \in [m], j \in [n] \tag{8}$$

The program defined by constraints (5), (7) and (8) is a binary quadratic program. We linearize this by introducing variables $y_{i,k,j,l} := x_{i,k} x_{j,l}$ and constraints

$$y_{i,k,j,l} \leq x_{i,k} \qquad \forall i \in [m], k \in [n], j \in \{i+1, \ldots, m\}, l \in \{k+1, \ldots, n\} \tag{9}$$

$$y_{i,k,j,l} \leq x_{j,l} \qquad \forall i \in [m], k \in [n], j \in \{i+1, \ldots, m\}, l \in \{k+1, \ldots, n\} \tag{10}$$

$$y_{i,k,j,l} \geq 0 \qquad \forall i \in [m], k \in [n], j \in \{i+1, \ldots, m\}, l \in \{k+1, \ldots, n\} \tag{11}$$

We have the following integer linear program.

$$
\begin{aligned}
\max_{\mathbf{x}} \quad & \sum_{i=1}^{m} \sum_{k=1}^{n} \sum_{j=i+1}^{m} \sum_{l=k+1}^{n} c_{i,k,j,l} y_{i,k,j,l} & \\
\text{s.t.} \quad & \sum_{(i,k) \in I} x_{i,k} \leq 1 & \forall I \in \mathcal{I} \\
& y_{i,k,j,l} \leq x_{i,k} & \forall i \in [m], k \in [n], j \in \{i+1, \ldots, m\}, l \in \{k+1, \ldots, n\} \\
& y_{i,k,j,l} \leq x_{j,l} & \forall i \in [m], k \in [n], j \in \{i+1, \ldots, m\}, l \in \{k+1, \ldots, n\} \\
& y_{i,k,j,l} \geq 0 & \forall i \in [m], k \in [n], j \in \{i+1, \ldots, m\}, l \in \{k+1, \ldots, n\} \\
& x_{i,j} \in \{0, 1\} & \forall i \in [m], j \in [n]
\end{aligned}
$$

Observe that there are an exponential number of constraints. Thus implementing this ILP by fully specifying all constraints will not scale in practice. Can we do better than this? Yes, we will separate these constraints as we solve the ILP. Initially, we omit constraints (7). We then solve the linear programming (LP) relaxation, where constraints (8) have been replaced by $0 \leq x_{i,k} \leq 1$. The resulting fractional solution $\bar{\mathbf{x}}$ may violate omitted constraints (7). How can we identify violated constraints?

As previously described, each maximal pairwise incompatible set I is a decreasing path in G from $(m, 1)$ to $(1, n)$. The task now is to find a longest path P in G between these two vertices, where the length of P is simply the sum of the fractional variables $\bar{x}_{i,k}$ of each vertex $v_{i,k}$ in P. We can do this using dynamic programming in $O(mn)$ time. If the longest path has length greater than 1 then we identified a violated constraint, which we add to the model. Otherwise, there are no violated constraints of the form (7).

If we identified violated constraints, we solve the updated model. This is repeated until there are no violated constraints. In this case $\bar{\mathbf{x}}$ might still be fractional. That is why we will use branching, where we pick a vertex $v_{i,k}$ and consider the two cases $x_{i,k} = 0$ and $x_{i,k} = 1$. For each branching case, we will solve the LP relaxation by separating cutting planes (as described above). This procedure is known as branch-and-cut (B&C), and enables one to solve practical problem instances of complicated combinatorial optimization problems.

References

[1] Alberto Caprara, Robert Carr, Sorin Istrail, Giuseppe Lancia, and Brian Walenz. 1001 Optimal PDB Structure Alignments: Integer Programming Methods for Finding the Maximum Contact Map Overlap. *Journal of Computational Biology*, 11(1):27–52, January 2004.

[2] Inken Wohlers. *Exact algorithms for pairwise protein structure alignment.* PhD thesis, VU University Amsterdam, 2012.