Incorporating $Ab\ Initio$ energy into threading approaches for protein structure prediction

Mingfu Shao[1], Sheng Wang[2], Chao Wang[1], Xiongying Yuan[1], Shuai Cheng Li[3] , Weimou Zheng [*2] and Dongbo Bu[*1]

[1]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing, China
[3]International Computer Science Institute, Berkeley

Email: Mingfu Shao - shaomingfu@ict.ac.cn; Sheng Wang - wangsheng@itp.ac.cn; Chao Wang - wangchao1987@ict.ac.cn; Xiongying Yuan - yuanxiongying@ict.ac.cn; Shuai Cheng Li - scli@icsi.berkeley.edu; Weimou Zheng - zheng@itp.ac.cn ; Dongbo Bu - dbu@ict.ac.cn;

[*]Corresponding author

Abstract

Background: Native structures of proteins are formed essentially due to the combining effects of $local$ and $distant$ (in the sense of sequence) interactions among residues. These interaction information are, explicitly or implicitly, encoded into the scoring function in protein structure prediction approaches—threading approaches usually measure an alignment in the sense that how well a sequence adopts an existing structure; while the energy function in $Ab\ Initio$ methods are designed to measure how likely a conformation is near-native. Encouraging progress has been observed in structure refinement where knowledge-based or physics-based potentials are designed to capture $distant$ interactions. Thus, it is interesting to investigate whether $distant$ interaction information captured by the $Ab\ Initio$ energy function can be used to improve threading, especially for the weakly/distant homologous templates.

Results: In this paper, we investigate the possibility to improve alignment-generating through incorporating $distant$ interaction information into the alignment scoring function in a nontrivial approach. Specifically, the distant interaction information is introduced through employing an $Ab\ Initio$ energy functions to evaluate the "partial" decoy built from an alignment. Subsequently, a local search algorithm is utilized to optimize the scoring function.

Experimental results demonstrate that with distant interaction items, the quality of generated alignments are improved on 68 out of 127 query-template pairs in Prosup benchmark. In addition, compared with state-to-art threading methods, our method performs better on alignment accuracy comparison.

Conclusions: Incorporating $Ab\ Initio$ energy functions into threading can greatly improve alignment accuracy.

Introduction

Protein structure determination is critical for understanding protein functions, and also highly relevant with therapeutics and drugs design. Computational prediction methods for protein structure plays important roles due to the speed of experimental determination methods cannot catch up with that of generation of protein primary sequences by genome projects. Computational protein structure prediction methods can be categorized into free modeling (FM) and template-based modeling (TBM). Specifically, for the protein without structural analogs in the template database, the structural conformation has to be built from the scratch; while for the proteins having structural analogs, the key step is to identify an accurate alignment between the query sequence and a template with known structure.

Both *Ab Initio* and threading approaches employ scoring functions to capture interactions among residues in an explicit or implicit manner. In essence, protein folding is the combining effects of local interactions and distant interactions among residues. Specifically, local interactions lead to local structural motifs, while nonlocal interactions arrange local structural motif structural fragments to form native-like structures.

The *Ab Initio* approaches for free modeling attempt to find a structural conformation with the lowest energy. Typically, local interactions are described via short structural fragments while nonlocal interactions are captured via an energy function. Various energy functions [1–5] have been proposed, and can be categorized into two classes, i.e., knowledge-based and physics-based. Compared with physics-based energy function, knowledge-based energy functions are more attractive since they are easy to use and understand. In addition, distance-dependent potentials perform better than distance-independent ones [6].

A typical template-based modeling procedure consists of a threading step to align the target protein onto a template, and a refinement step to refine the template structure to be more native-like. Numerous threading methods have been proposed to calculate the optimal alignments under different scoring functions. These threading methods can be categorized into the following classes based on the divergence of scoring functions:

1. The scoring function does not contain any non-local interaction information explicitly. For example, FASTA [7], BLAST [8], and PSI-BLAST [9] assume independence among residues at different positions while HMMer [10] and HHpred [11] apply Hidden Markov Model to introduce the transition information between adjacent residues into scoring function. Since only local information are taken into consideration in their scoring functions, dynamic programming is a natural technique to obtain a global optimal solution.

2. The scoring function captures nonlocal interaction information via contact preference. That is, if a pair of residues in the query sequence are aligned to the two ends of an interaction, then this pair will be given a score according to a contact preference matrix. PROSPECT [12] and RAPTOR [13] implemented this kind of energy function and demonstrated the improvements of prediction accuracy. However, the following features of nonlocal interactions were not taken into consideration explicitly: (i) it is more accurate to describe pairwise interactions in distance-dependent manner than distance-independent ways; and (ii) besides distance, the orientation angles involved in dipole–dipole interactions have also been proved to be useful to discriminate native structures.

The purposes of the study is to investigate whether threading results can be improved through incorporating *Ab Initio* energy function. Distant interactions are usually described in a more accurate manner in *Ab Initio* energy function. For example, dDFIRE employs distance-dependent pairwise interaction rather than distance-independent one. Encouraging progress has been observed in structure refinement where *Ab Initio* energy function is employed to refine template structure to be more native-like. It is interesting whether *Ab Initio* energy function improves alignment generating.

In addition, when the global structural information is incorporated, effective algorithms such as dynamic programming do not work any more: if all pairwise interactions are added into scoring function, the optimization problem becomes NP-hard [14]. A variety of techniques, such as integer linear programming [13] and divide and conquer [15] have been proposed to solve this problem. In this study, we propose an efficient, local search based method to identify optimal alignments. Comparing with existing

methods [13, 15], which are designed specifically for scoring functions consisting of distant-independent pairwise interaction alone as their global item, our method is more general and can be used to optimize any kind of scoring functions.

Scoring model

The scoring function to assess an alignment $\mathcal{A}$ consists of local item $L(\mathcal{A})$ and distant item $G(\mathcal{A})$, i.e., $score(\mathcal{A}) = \omega_L L(\mathcal{A}) + G(\mathcal{A})$, where ω_L denotes weight of local item.

Local item is the weighted sum of mutation score S_m, secondary structure compatibility score S_{ss}, solvent accessibility score S_{sa}, gap penalty score S_g, and structural segment compatibility score S_{CLE} [16], i.e., $L(\mathcal{A}) = \omega_m S_m(\mathcal{A}) + \omega_{ss} S_{ss}(\mathcal{A}) + \omega_{CLE} S_{CLE}(\mathcal{A}) + \omega_{sa} S_{sa}(\mathcal{A}) + \omega_g S_g(\mathcal{A})$, $S_g(\mathcal{A}) = \omega_{go} GO + \omega_{ge} GE$, where GO and GE are the number of gap open and gap extending, respectively. The weight of these items are to be determined via training on SALIGN benchmark.

The global item $G(\mathcal{A})$, which contains the nonlocal interaction information implicitly, is captured by the dDFIRE energy over a "partial" decoy corresponding to the alignment $\mathcal{A}$. An ideal way to measure nonlocal interaction is to calculate dDFIRE energy over a full-length decoy. However, it is usually time-consuming to obtain full-length decoy through running structure-generating tools such as MODELLER [17]. Thus, this strategy is unacceptable since we usually need to sample thousands of alignments. Here, we employ an alternative method to build a partial "decoy" from the alignment. Specifically, only the aligned residues are kept with their coordinates simply copied from the corresponding residues in the template.

This section are organized as follows: We first verify that dDFIRE energy function is constantly good-performing when used to evaluate "partial" decoys. Second, both local item and global item should be normalized using match state size. Third, we prove that global item of the our scoring function is effective to capture distance interaction comparing with contact-preference based scoring functions. Fourth, we show that optimal local score can be used to determine "easy" pairs for which local score item is sufficient while adding global item may lead noise contrarily. Last, we train ω_L on SALIGN [18] benchmark dataset.

Performance of dDFIRE on partial structure

Since we calculate dDFIRE energy on the "partial" stead of a full-length structure, thus it is necessary to verify whether the "partial dDFIRE energy" still have the power to distinguish native-like decoys. To verify this, we performed experiments on three commonly-used benchmark datasets: LKF [1], Gapless Threading [2] and Rosetta [5]. The datasets contain 178, 200 and 232 proteins, respectively; and for each protein, 100 decoys were generated as control to the native structure. The objective of this experiment is to verify whether the "partial" native structure can be distinguished from the "partial" decoys by dDFIRE.

For both native structures and decoys, the "partial" conformations were simulated through randomly excising a set of residues. At various excising percentage, the ratio of proteins for which the partial native structure has the lowest dDFIRE energy relative to all partial decoys are calculated, and denoted as accuracy in Fig.1. As demonstrated by Fig.1, on LKF and Rosetta benchmarks, dDFIRE performs constantly well even if over 40% residues are excised; and on Gapless Threading benchmark, the performance decreases slightly.

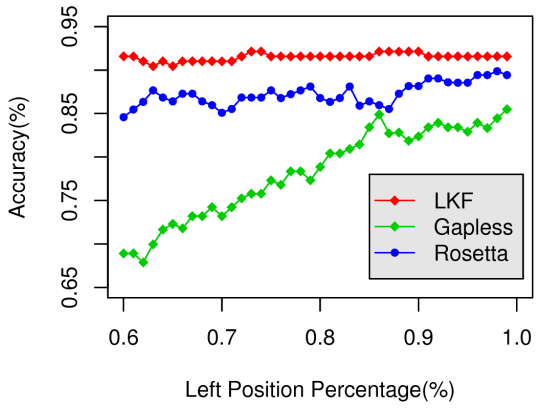

Figure 1: Performance of dDFIRE to distinguish "partial" native structure from "partial" decoys. X-axis is the ratio of remaining residues after the excising process, and Y-axis denotes the ratio of proteins for which the "partial" native structure still have lower energy than "partial" decoys.

Score normalization

We also investigate the relationship between the scores with the match state size. Analysis suggests

the linearity between local(global) scores and match state size. Specifically, the linear correlation coefficient between local(global) scores and the match state size is -0.762 (-0.968) (See Fig.2 and 3 for details). Thus, it is reasonable to normalize both local and global score through dividing by the match state size.

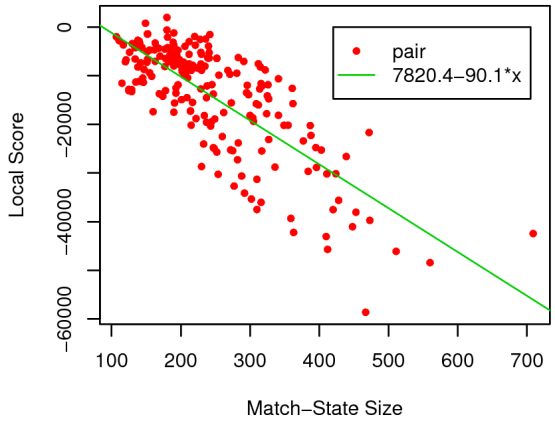

Figure 2: Linear correlation between local score and match state size. Both local score and match state size are calculated from reference alignment of query-template pairs in SALIGN [18] benchmark dataset.

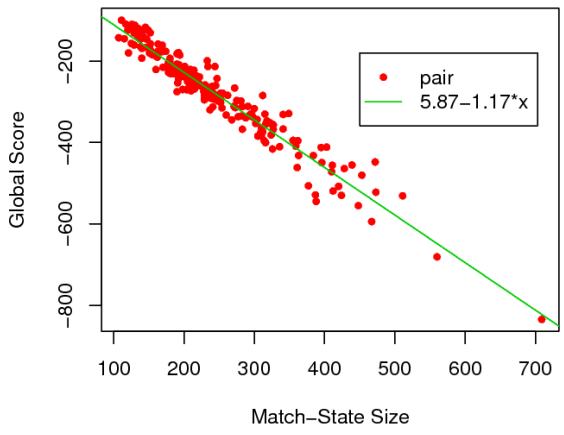

Figure 3: Linear correlation between global score and match state size.

Effect of global items

We further investigate the effect of global item. As control, we performed comparison with the traditional way to describe nonlocal interactions via contact preference matrix [13, 15], i.e, $S_p = \sum_i \sum_j \delta(i,j) Pair(\mathcal{A}(i), \mathcal{A}(j))$, and $Pair(m,n) = P_m^T C P_n$ where $\mathcal{A}(i)$ is the matched residue in the sequence, $\delta(i,j)$ indicates whether ith and jth residue in the template have contact, P_m is the profile vector at the mth position and C is the contact preference matrix.

We first give some notations before presenting the experiments to examine the effects of global items. For each query-template pair, two typical alignments are generated: the structural alignment $\mathcal{A}_R$ generated via running TMalign [19], and the optimal alignment (denoted as $\mathcal{A}_L$) when only local item $L(\mathcal{A})$ is taken into consideration, i.e. $\mathcal{A}_L = \arg\min_{\mathcal{A}} L(\mathcal{A})$. For each alignment $\mathcal{A}$, its real quality is measured by TMscore [20], denoted as $\mathrm{TM}(\mathcal{A})$. We also use $L(\mathcal{A})$ and $G(\mathcal{A})$ to denote the local score and global score of $\mathcal{A}$, and use $C(\mathcal{A})$ to denote the contact-preference-based score of $\mathcal{A}$.

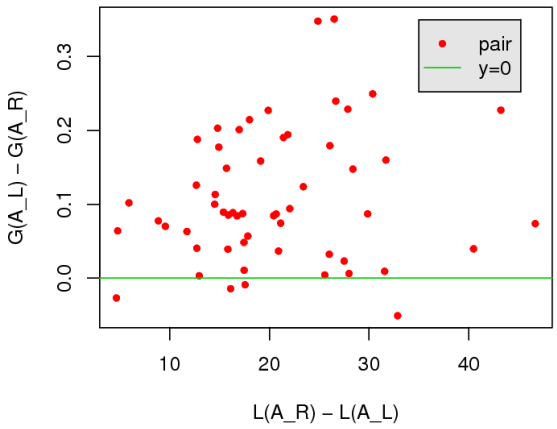

Figure 4: Effect of global score to distinguish $\mathcal{A}_R$ from $\mathcal{A}_L$. All points lies to the right of $x = 0$, and 52 of 56 points appear above $y = 0$.

The 200 query-template pairs in SALIGN [18] dataset are categorized into two classes according to the quality of $\mathcal{A}_L$: (i) $\mathrm{TM}(\mathcal{A}_R) - \mathrm{TM}(\mathcal{A}_L) < 0.1$, 144 pairs in total; and (ii) $\mathrm{TM}(\mathcal{A}_R) - \mathrm{TM}(\mathcal{A}_L) \geq 0.1$, 56 pairs in total. Intuitively, class 1 contains the pairs for which a scoring function with local score item alone is sufficient; and class 2 contains the pairs for which local score alone failed. For pairs in class 2, we expect global items can help to distinguish the reference alignment. We verify this by comparing the global score of $\mathcal{A}_L$ and $\mathcal{A}_R$: only for pairs sat-

isfying $\mathcal{A}_L$ - $\mathcal{A}_R > 0$, it is likely to distinguish the reference alignment.

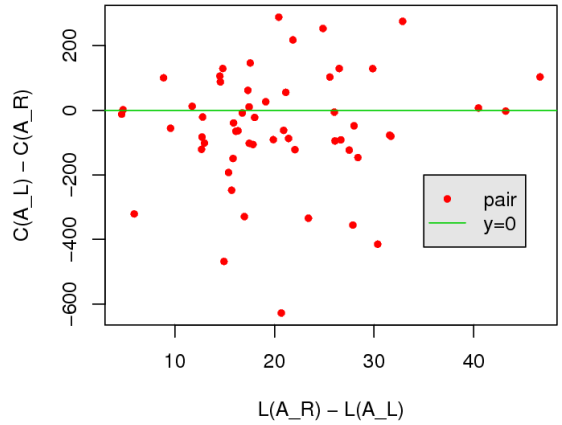

Figure 5: Effect of contact-preference-based score to distinguish $\mathcal{A}_R$ from $\mathcal{A}_L$. Only 20 of 56 points appear above $y = 0$.

Fig.4 and 5 suggest that for the pairs that local item alone cannot separate $\mathcal{A}_L$ from $\mathcal{A}_R$ ($L(\mathcal{A}_L) \leq L(\mathcal{A}_R)$ because of $\mathcal{A}_L = \arg\min_{\mathcal{A}} L(\mathcal{A})$), global item of our scoring function can effectively measure the quality of alignments. Specifically, we observed that $G(\mathcal{A}_R) < G(\mathcal{A}_L)$ on 52 of 56 pairs. In contrast, the contact-preference-based score does not help improve this situation, only on 20 of 56 pairs, $C(\mathcal{A}_R) < C(\mathcal{A}_L)$.

Determining pairs for which local scores alone is sufficient

On 144 of 200 pairs of SALGIN benchmark, local score alone is sufficient to find out a "good" alignment($\mathrm{TM}(\mathcal{A}_R) - \mathrm{TM}(\mathcal{A}_L) < 0.1$). In fact, in these cases, adding global item may lead to false-negative [21]. We observed that the normalization of local scores help recognizing these "easy" pairs. This is reasonable since local score contains most of the homologous information between the sequence and the template.

Fig.6 implies that TMscore value is strongly correlated with local score (linear correlation coefficient is -0.78). Besides, as the local score increase, $\mathcal{A}_L$ becomes worse, i.e., the cumulative average value of $\mathrm{TM}(\mathcal{A}_R) - \mathrm{TM}(\mathcal{A}_L)$ increases as the local score increasing (the blue curve in Fig.6). Accordingly, we choose a threshold of local score, denoted as θ, to determine whether local score item is sufficient: if $L(\mathcal{A}_L) \leq \theta$, then $\mathcal{A}_L$ is treated as a good alignment. In our method, $\theta = -87$.

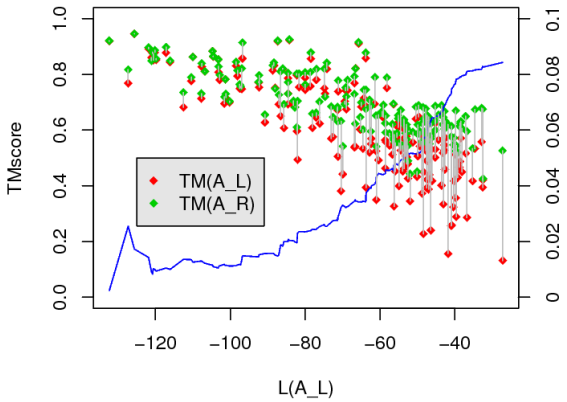

Figure 6: Linear correlation between TMscore and normalized local score. For each pair in SALIGN benchmark dataset, TMscore of reference alignment(green points) and TMscore of $\mathcal{A}_L$ are compared with local score of $\mathcal{A}_L$. Length of gray segment represents the difference of TMscore. The average difference of TMscore(using right axis) along with local score increasing is showed as the blue line.

Weight training process

Parameter ω_L is trained by classification. For each query-template pair in SALIGN benchmark, one positive alignment $\mathcal{A}_p$ and 10 negative alignments $\mathcal{A}_n$ are selected (We also have tried other number of negative alignments, similar result is obtained).

Here, we use the reference alignment as positive alignment, i.e. $\mathcal{A}_p = \mathcal{A}_R$. Negative alignments are chosen from the top 1000 alignments returned by dynamic programming. We first cluster these alignments to remove redundancy, and then randomly select alignments satisfying $\mathrm{TM}(\mathcal{A}_p) - \mathrm{TM}(\mathcal{A}_n) > 0.2$.

ω_L should divide $\mathcal{A}_n$ and $\mathcal{A}_p$ as much as possible. Formally

$$\omega_L = \arg\max |H|$$

where

$$\begin{aligned} H &= \{(\mathcal{A}_p, \mathcal{A}_n) \in \mathcal{P} | \omega_L L(\mathcal{A}_p) + G(\mathcal{A}_p) < \omega_L L(\mathcal{A}_n) + G(\mathcal{A}_n)\} \\ &= \{(\mathcal{A}_p, \mathcal{A}_n) \in \mathcal{P} | \omega_L (L(\mathcal{A}_p) - L(\mathcal{A}_n)) < G(\mathcal{A}_n) - G(\mathcal{A}_p)\}. \end{aligned}$$

$\mathcal{P} = \{(\mathcal{A}_p, \mathcal{A}_n) | \mathcal{A}_p$ and $\mathcal{A}_n$ are from the same pair$\}$, since alignments of different query-template pair are not comparable. The classification result is showed in Fig.7, $\omega_L = 0.0047$.

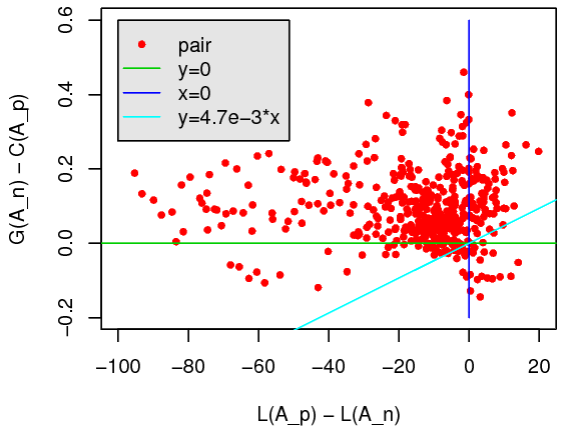

Figure 7: Training ω_L on SALIGN benchmark. X-axis is $L(\mathcal{A}_p) - L(\mathcal{A}_n)$ while Y-axis is $G(\mathcal{A}_n) - G(\mathcal{A}_p)$. The optimization problem requires a positive-slope line with the most points above it.

After obtaining the parameter ω_L and θ, our threading algorithm can be described informally as follows: given a query-template pair, dynamic programming algorithm is employed to calculate $\mathcal{A}_L$. If $L(\mathcal{A}_L) < \theta$, then $\mathcal{A}_L$ is considered as a good alignment and returned. Otherwise, local search algorithm is then used to find a better alignment under scoring function $score(\mathcal{A}) = \omega_L L(\mathcal{A}) + G(\mathcal{A})$. The initial alignments used in this step are chosen from the dynamic programming table in the previous step.

Preliminary results on alignment generating

We test our threading method on Prosup benchmark (containing 127 query-template pairs) Each query-template pair shares low sequence identity but high structure similarity. Denote the alignment generated by our method as $\mathcal{A}_O$.

First, we compare TM($\mathcal{A}_O$) with TM($\mathcal{A}_L$) in order to evaluate the effect of the new scoring function. The result is showed in Fig.8. It suggests that on 68 out of 127 pairs the new scoring function gain a better TMscore compared with scoring function with local item only. On 12 out of 127 pairs, TMscore improvement is greater than 0.1 while no pair's TMscore decrease greater than 0.1.

Second, we compare the alignment accuracy with other threading methods. For an alignment, its accurate accuracy is defined as the ratio of number of correct match-state over the number of match-state of the reference alignment; the ratio is denoted as ±4-residues-accuracy if a ±4 error allowed. Experimental results(Table 1) indicate that our method performs better than FASTA, Sequence and PSI-BLAST. If only the local score item is considered, the alignment accuracy is comparable to RAPTOR. When the distant scoring item is added, the alignment accuracy improves significantly: 8% better than RAPTOR on accurate comparison and 6.4% on ±4-residues comparison.

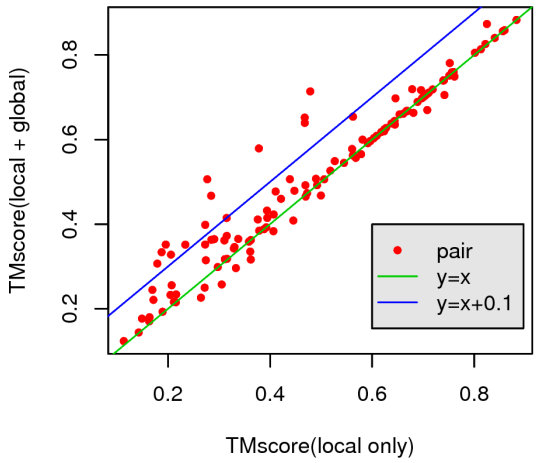

Figure 8: TMscore comparison between TM($\mathcal{A}_L$) and TM($\mathcal{A}_O$) on Prosup benchmark. X-axis is TM($\mathcal{A}_L$) while Y-axis is TM($\mathcal{A}_O$). Each red point is a pair in Prosup benchmark. Green line is $y = x$. Points above this line represents the new scoring function has a better performance. Blue line is $y = x + 0.1$. Points above this line represents an improvement over 0.1. 20 points are above blue line.

Methods	Accurate(%)	±4-residues(%)
FASTA	31.4	-
Sequence	34.1	-
PSI-BLAST	35.6	-
RAPTOR	44.0	63.7
$\mathcal{A}_L$	43.1	63.0
$\mathcal{A}_O$	**52.0**	**70.1**

Table 1: Alignment Accuracy Comparison on Prosup Benchmark. Result of FASTA and Sequence are from [22]; result of PSI-BLAST is from [23]; result of RAPTOR is obtained from the binary version running by us, this result may not reflect the accuracy of the current version.

Methods

Threading Algorithm

The framework of our threading algorithm are described as follows:

Algorithm 1(Threading Algorithm)
Input: query sequence, template, θ, ω_L, α, k
Output: an alignment between query and template

step 1 set $score(\mathcal{A}) = L(\mathcal{A})$, calculate the optimal alignment $\mathcal{A}_L$ under this scoring function by dynamic programming algorithm, save the best 100 alignments from the dynamic programming table

step 2 calculate $L(\mathcal{A}_L)$, if $L(\mathcal{A}_L) < \theta$, then return $\mathcal{A}_L$

step 3 set $score(\mathcal{A}) = \omega_L L(\mathcal{A}) + G(\mathcal{A})$, for each alignment $\mathcal{A}_i$ in the 100 candidates in step 1, run local search algorithm(Algorithm 2 described in the following subsection) with parameter α, k and initialize alignment $\mathcal{A}_i$, it returned $\mathcal{A}_{Oi}$

step 4 return $\arg\min_{i=1}^{100} \omega_L L(\mathcal{A}_{Oi}) + G(\mathcal{A}_{Oi})$

Local Search Algorithm

In this sub-section we describe the threading problem in a concise way, propose a local search algorithm based on a new neighborhood for general scoring function. Under a certain assumption, we prove its approximation guarantee for two specific scoring functions.

Problem Formulation

We first give some formal definitions.

Definition 1. *Given a template* $T = \{t_1, t_2, \cdots, t_m\}$, $t_i < t_{i+1}$ *and a sequence* $S = \{s_0, s_1, s_2, \cdots, s_n\}$, $s_i < s_{i+1}$, *a valid* alignment *is a non-decreasing mapping* $\mathcal{A}$ *from* T *to* S.

Denote all valid alignments as $\mathcal{F}$. Non-decreasing mapping is equivalent with traditional alignment definition with *gap*. For all t satisfying $\mathcal{A}(t) = s$, $s > s_0$, we can define the smallest t actually matches with s while others are gap on template. In order to allow gap on the left end of the template, we add a extra amino acid s_0 in the left end of the sequence. All $t \in T$ aligned to s_0 are gap on the left end of the template. Mapping allows gap on sequence naturally.

Now we define the neighborhood of an alignment. Denote the k-neighbor of $\mathcal{A}$ as $N(\mathcal{A}, k)$, we have the following definition.

Definition 2. *Suppose* $\mathcal{A}' \in \mathcal{F}$, *then* $\mathcal{A}' \in N(\mathcal{A}, k)$ *if and only if there exists a subset* U *of* S, *satisfying* $|U| \leq k$ *and* $\forall t \in T$ $\mathcal{A}'(t) \in \{\mathcal{A}(t)\} \cup U$.

Intuitively, a member of k-neighbors of $\mathcal{A}$ differs with $\mathcal{A}$ only on at most k positions at sequences.

Claim 1. $N(\mathcal{A}, k) \subset N(\mathcal{A}, k+1)$.

Claim 2. $N(\mathcal{A}, n+1) = \mathcal{F}$.

Claim 3. $|N(\mathcal{A}, k)| = O(m^{2k} n^k), \forall \mathcal{A} \in \mathcal{F}$.

Claim 1 and claim 2 are obvious. The proof claim 3 is put in the Appendix.

Claim 1 and claim 2 shows that with the increasing of k, the number of neighbors of an alignment is growing and eventually reaches the whole space. Claim 3 estimate the size of $|N(\mathcal{A}, k)|$. It shows that for a fixed k the number of neighbors of a valid alignment is polynomial about m and n.

Definition 3. *For any* $\mathcal{A} \in \mathcal{F}$, *there is a real positive number denoted as* $score(\mathcal{A})$ *to evaluate* $\mathcal{A}$, *the* threading problem *is* $\min_{\mathcal{A} \in \mathcal{F}} score(\mathcal{A})$.

$score(\mathcal{A})$ is the general representation of scoring function. In this study, $score(\mathcal{A}) = \omega_L L(\mathcal{A}) + G(\mathcal{A})$.

Algorithm

Based on the definition of neighborhood above, we give the local search algorithm as follows:

Algorithm 2(Local Search)
Input $\alpha \geq 0$, k, initial alignment $\mathcal{A}_0$
Output an approximate local optimal solution of the scoring function $score(\mathcal{A})$

step 1 $i = 0$, initialize $\mathcal{A}_0$ according to input

step 2 calculate $\mathcal{A}_{i+1} = \arg\min_{\mathcal{A} \in N(\mathcal{A}_i, k)} score(\mathcal{A})$

step 3 if $(1+\alpha) score(\mathcal{A}_{i+1}) < score(\mathcal{A}_i)$, $i = i+1$, goto *step 2*

step 4 output $\mathcal{A}_i$

When $\alpha = 0$, we can obtain an accurate local optimal solution. When $\alpha = 0$ and $k = n+1$, we can obtain an accurate global optimal solution.

Claim 4. *Suppose $\alpha > 0$, The time complexity of algorithm 2 is $O(m^{2k}n^k \log_{1+\alpha} M)$, where $M = \frac{\sup_{\mathcal{A}\in\mathcal{F}} score(\mathcal{A})}{\inf_{\mathcal{A}\in\mathcal{F}} score(\mathcal{A})}$*

Proof. Based on the algorithm, we have

$$(1+\alpha)^i score(\mathcal{A}_i) < score(\mathcal{A}_0),$$

which implies that

$$i < \log_{1+\alpha} \frac{score(\mathcal{A}_0)}{score(\mathcal{A}_i)} \leq \log_{1+\alpha} M.$$

According to claim 3, each iteration wastes at most $O(m^{2k}n^k)$ time, so the claim is proved. □

If a *closing assumption* is satisfied, we can prove two approximation guarantee results when the scoring function only consist of local item and pairwise contact item. Details are listed in the Appendix section.

Discussion

In order to employ general energy function, the key step is transforming alignment to decoy efficiently. In this paper, "partial" decoy strategy is quick enough but not accurate because only matched residues' backbone and C_β atoms are kept. Methods that effectively recover other unmatched residues and even side chain atoms according to alignments are imperative.

Though the energy function of *Ab Initio* can be used by threading, the two methods have fundamental difference on the divergence of search space. Actually, the search space of threading is much smaller than that of *Ab Initio* methods because many useful prior knowledge can greatly narrow its search space. For instance, we can restrict that a core on template either totally aligned or totally gaped. This prior has been verified and applied by many threading methods. Consequently, the search space can be reduced to $O(N^m)$ where N is the number of cores and m the length of query sequence. On average, $N \leq 10$, it is much smaller than 200^m, which is the search space of ROSETTA.

In this paper we have proposed a local search algorithm to find out the optimal solution of general scoring function. This algorithm is based on a neighborhood definition, and this neighborhood can also be used by other search strategies such as simulated annealing and genetic algorithm.

References

1. Loose C, Klepeis J, Floudas C: **A new pairwise folding potential based on improved decoy generation and side-chain packing**. *Proteins: Structure, Function, and Bioinformatics* 2004, **54**(2):303–314.
2. Zhang J, Chen R, Liang J: **Empirical potential function for simplified protein models: Combining contact and local sequence–structure descriptors**. *Proteins: Structure, Function, and Bioinformatics* 2006, **63**(4):949–960.
3. Ranjit B, Pinak C: **Discriminating the native structure from decoys using scoring functions based on the residue packing in globular proteins**. *BMC Structural Biology***9**.
4. Shen M, Sali A: **Statistical potential for assessment and prediction of protein structures**. *Protein Science* 2006, **15**(11):2507–2524.
5. Simons K, Kooperberg C, Huang E, Baker D: **Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions1**. *Journal of Molecular Biology* 1997, **268**:209–225.
6. Zhou H, Zhou Y: **Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction**. *Protein Science* 2002, **11**(11):2714–2726.
7. Pearson W, Lipman D: **Improved tools for biological sequence comparison**. *Proceedings of the National Academy of Sciences of the United States of America* 1988, **85**(8):2444.
8. Altschul S, Gish W, Miller W, Myers E, Lipman D: **Basic local alignment search tool**. *Journal of molecular biology* 1990, **215**(3):403–410.
9. Altschul S, Madden T, Schaffer A, Zhang J, Zhang Z, Miller W, Lipman D: **Gapped BLAST and PSI-BLAST: a new generation of protein database search programs**. *Nucleic acids research* 1997, **25**(17):3389.
10. Durbin R: *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge Univ Pr 1998.

11. Soding J: **Protein homology detection by HMM–HMM comparison**. *Bioinformatics* 2005, **21**(7):951.

12. Xu Y, Xu D, Uberbacher E: **An Efficient Computational Method for Globally Optimal Threading1**. *Journal of Computational Biology* 1998, **5**(3):597–614.

13. Xu J, Li M, Kim D, Xu Y: **RAPTOR: optimal protein threading by linear programming**. *INTERNATIONAL JOURNAL OF BIOINFORMATICS AND COMPUTATIONAL BIOLOGY* 2003, **1**:95–118.

14. Lathrop R: **The protein threading problem with sequence amino acid interaction preferences is NP-complete**. *Protein Engineering Design and Selection* 1994, **7**(9):1059.

15. Xu Y, Xu D: **Protein threading using PROSPECT: design and evaluation**. *Proteins: Structure, Function, and Bioinformatics* 2000, **40**(3):343–354.

16. Wang S, Zheng W: **CLePAPS: fast pair alignment of protein structures based on conformational letters**. *Journal of bioinformatics and computational biology* 2008, **6**(2):347–366.

17. Eswar N, Webb B, Marti-Renom M, Madhusudhan M, Eramian D, Shen M, Pieper U, Sali A: **Comparative protein structure modeling using MODELLER**. *Curr Protoc Protein Sci* 2007.

18. Marti-Renom M, Madhusudhan M, Sali A: **Alignment of protein sequences by their profiles**. *Protein Science* 2004, **13**(4):1071–1087.

19. Zhang Y, Skolnick J: **TM-align: a protein structure alignment algorithm based on the TM-score**. *Nucleic acids research* 2005, **33**(7):2302.

20. Zhang Y, Skolnick J: **Scoring function for automated assessment of protein structure template quality**. *Proteins: Structure, Function, and Bioinformatics* 2004, **57**(4):702–710.

21. Peng J, Xu J: **Boosting protein threading accuracy**. In *Research in Computational Molecular Biology*, Springer 2009:31–45.

22. Domingues F, Lackner P, Andreeva A, Sippl M: **Structure-based evaluation of sequence comparison and fold recognition alignment accuracy1**. *Journal of molecular biology* 2000, **297**(4):1003–1013.

23. Zhou H, Zhou Y: **Fold recognition by combining sequence profiles derived from evolution and from depth-dependent structural alignment of fragments**. *Proteins: Structure, Function, and Bioinformatics* 2005, **58**(2):321–328.

Appendix

Proof of Claim 3

Proof. There is $\binom{n}{k} = O(n^k)$ cases to choose a subset U of S while $|U| = k$. Denote the neighbors of an alignment under a certain U as $N_U(\mathcal{A})$. So, we only need to prove $|N_U(\mathcal{A})| = O(m^{2k})$.

We can assume that all positions in U are not aligned, that is, there exists no $t \in T$ such that $\mathcal{A}(t) = u_i$. If not, say, u_i is aligned, we can extend S to $S' = \{s_0, s_1, \cdots, s_j = u_i, u_i', s_{j+1}, \cdots s_n\}$ and change U to $U' = \{u_1, \cdots, u_{i-1}, u_i', u_{i+1}, \cdots, u_k\}$. Obviously, $|N_U(\mathcal{A})| \leq |N_{U'}(\mathcal{A})|$.

Consider the sub-problem when $T_i = \{t_1, t_2, \cdots, t_i\}$, $U_j = \{u_1, u_2, \cdots, u_j\}$, $1 \leq i \leq m$ and $1 \leq j \leq k$. For this sub-problem, we define $A(i,j) = \{g \in N_{U_j}(\mathcal{A}) | g(t_i) = u_j\}$, and $B(i,j) = \{\mathcal{A}' \in N_{U_j}(\mathcal{A}) | \mathcal{A}'(t_i) = \mathcal{A}(t_i)\}$. Then $|N_U(\mathcal{A})| = |A(m,k)| + |B(m,j)|$.

Now we give out the iterative formula. Let $a_i = \inf\{j | \mathcal{A}(t_j) \geq u_i\}$. then $a_i \leq a_{i+1}$. Without losing generality, in the following prove, we assume that $a_i \leq a_{i+1}$. Define $\delta(x) = 1$ when $x \geq 0$ and $\delta(x) = 0$ when

$x < 0$, we have

$$\begin{aligned} |A(i,j)| &= \sum_{l=1}^{j} |A(i-1,l)| + \delta(a_j - i)|B(i-1,j)| \\ |B(i,j)| &= \sum_{l=1}^{j} \delta(i-a_l)|A(i-1,l)| + |B(i-1,j)| \end{aligned}$$

we employ mathematics induction to prove:

$$\begin{array}{lcll} |A(i,j)| & \leq & i^j & \text{if} \quad 1 \leq i \leq a_1 \\ |A(i,j)| & \leq & i^{l+j} & \text{if} \quad a_l < i \leq a_{l+1}, 1 \leq l < j \\ |A(i,j)| & \leq & i^{2j-1} & \text{if} \quad i > a_j \\ |B(i,j)| & \leq & 1 & \text{if} \quad 1 \leq i < a_1 \\ |B(i,j)| & \leq & i^{2l-1} & \text{if} \quad i = a_l, 1 \leq l \leq j \\ |B(i,j)| & \leq & i^{2l} & \text{if} \quad a_l < i < a_{l+1}, 1 \leq l < j \\ |B(i,j)| & \leq & i^{2j} & \text{if} \quad i > a_j. \end{array}$$

Firstly, $|B(1,j)| = |A(1,j)| = 1$.
When $1 < i < a_1$,

$$\begin{aligned} |B(i,j)| &= |B(i-1,1)| = 1 \\ |A(i,j)| &= \sum_{l=1}^{j} |A(i-1,l)| + |B(i-1,j)| \leq \sum_{l=1}^{j} (i-1)^l + 1 \\ &\leq \frac{1-(i-1)^{j+1}}{1-(i-1)} = \frac{(i-1)^{j+1}-1}{i-2} \leq i^j \end{aligned}$$

When $i = a_1$,

$$\begin{aligned} |B(i,j)| &= |A(i-1,1)| + |B(i-1,j)| = i \\ |A(i,j)| &= \sum_{l=1}^{j} |A(i-1,l)| + |B(i-1,j)| \leq i^j \end{aligned}$$

When $a_l < i < a_{l+1} \leq a_j$,

$$\begin{aligned} |B(i,j)| &= \sum_{p=1}^{l} |A(i-1,p)| + |B(i-1,j)| = \sum_{p=1}^{l} (i-1)^{p+l} + |B(i-1,j)| \\ &\leq \frac{(i-1)^{2l+1}}{i-2} + |B(i-1,j)| \leq \sum_{p=a_l}^{i-1} \frac{p^{2l+1}}{p-1} + |B(a_l,j)| \\ &\leq 2\int_{a_l}^{i} x^{2l} dx + a_l^{2l-1} \leq \frac{2(i^{2l} - a_l^{2l})}{(a_l - 2)(2l+1)} + a_l^{2l-1} \leq (\frac{2}{2l+1} + \frac{1}{i}) i^{2l} \leq i^{2l} \\ |A(i,j)| &= \sum_{p=1}^{j} |A(i-1,p)| + |B(i-1,j)| \leq \sum_{p=1}^{j} (i-1)^{p+j} + (i-1)^{2l} \\ &\leq \frac{(i-1)^{l+j+1}}{i-2} + (i-1)^{2l} \leq i^{l+j} \end{aligned}$$

Similar deduction can be used in the case of $i > a_j$. So the claim is proved. □

Approximation Guarantee of Local Search Algorithm

In this sub-section, we prove two approximation results under a certain assumption. The neighbor we used here is 1-neighbor. From claim 1 we know that for k-neighbor, $k > 1$, we can obtain a better result.

The algorithm's approximation guarantee is closely linked to the specific form of $score(\mathcal{A})$. First we only consider $score(\mathcal{A})$ consists of local items: $score(\mathcal{A}) = \sum_{t\in T} m(t, \mathcal{A}(t))$. For convenience's sake, we define the following marks.

$$\begin{aligned} T_i(\mathcal{A}) &= \{t \in T | \mathcal{A}(t) = s_i\} \\ \mathcal{A}(T') &= \{\mathcal{A}(t) | t \in T'\}, T' \subset T \\ m(T', \mathcal{A}) &= \sum_{t\in T'} m(t, \mathcal{A}(t)), T' \subset T \end{aligned}$$

Due to the technical reasons, we have to do a assumption. In the process of prove, we only need that local optimal solution and distant optimal solution satisfy the assumption, unfortunately, this does not always hold too.

Assumption 1. *Given 2 alignments $\mathcal{A}$ and g, define*

$$\mathcal{A}_i(t) = \begin{cases} \mathcal{A}(t), & \text{if } t \notin T_i(g) \\ s_i, & \text{otherwise} \end{cases} \qquad i = 0, 1, 2, \cdots, n$$

if $\mathcal{A}_i \in \mathcal{F}, i = 0, 1, 2, \cdots, n$, we say $\mathcal{A}$ and g satisfies closing assumption.

If the above assumption is satisfied, we have the following theorem.

Theorem 1. *If $score(\mathcal{A}) = \sum_{t\in T} m(t, \mathcal{A}(t)), \forall \mathcal{A} \in \mathcal{F}$, $\mathcal{A}^{**}$ is the distant optimal solution, $\mathcal{A}^*$ is the approximate local optimal solution obtained from algorithm 2 with factor α and $k = 1$, $\mathcal{A}^*$ and $\mathcal{A}^{**}$ satisfies closing assumption, n is the length of given sequence, $\alpha n < 1$. then*

$$score(\mathcal{A}^*) \leq \frac{1+\alpha}{1-\alpha n} score(\mathcal{A}^{**}).$$

Proof. Define

$$\mathcal{A}_i(t) = \begin{cases} \mathcal{A}^*(t), & \text{if } t \notin T_i(\mathcal{A}^{**}) \\ s_i, & \text{otherwise} \end{cases} \qquad i = 0, 1, 2, \cdots, n$$

$\mathcal{A}^*$ and $\mathcal{A}_i$ differs only in T_i(in this proof, we abbreviate $T_i(\mathcal{A}^{**})$ as T_i), so

$$score(\mathcal{A}^*) - score(\mathcal{A}_i) = m(T_i, \mathcal{A}^*) - m(T_i, \mathcal{A}^{**})$$

From the assumption, we know that $\mathcal{A}_i \in \mathcal{F}$, even more, $\mathcal{A}_i \in N_i(\mathcal{A}^*) \subset N(\mathcal{A}^*)$ which means $score(\mathcal{A}^*) \leq (1+\alpha) score(\mathcal{A}_i)$. Notice that $\cup_{i=0}^{n} T_i = T$ and $T_i \cap T_j = \emptyset, i \neq j$. We have,

$$\begin{aligned} & \sum_{i=0}^{n} \big(score(\mathcal{A}^*) - score(\mathcal{A}_i) - \alpha score(\mathcal{A}_i)\big) \\ = & \sum_{i=0}^{n} \big(m(T_i, \mathcal{A}^*) - (1+\alpha) m(T_i, \mathcal{A}^{**}) - \alpha m(T - T_i, \mathcal{A}^*)\big) \\ = & [1 - \alpha n] score(\mathcal{A}^*) - (1+\alpha) score(\mathcal{A}^{**}) \leq 0 \end{aligned}$$

which implicates the conclusion. □

Corollary 1. *If $score(\mathcal{A}) = \sum_{t\in T} m(t, \mathcal{A}(t)), \forall \mathcal{A} \in \mathcal{F}$, $\mathcal{A}^*$ is the accurate local optimal solution, then $score(\mathcal{A}^*) = score(\mathcal{A}^{**})$.*

Proof. This is the special case of theorem 1 when $\alpha = 0$. □

If pair contact is taken into scoring function: $score(\mathcal{A}) = \sum_{t\in T} m(t, \mathcal{A}(t)) + \sum_{u\in T}\sum_{v\in T} p(u, \mathcal{A}(u), v, \mathcal{A}(v))$ we have following theorem.

Theorem 2. *If* $score(\mathcal{A}) = \sum_{t\in T} m(t, \mathcal{A}(t)) + \sum_{u\in T}\sum_{v\in T} p(u, \mathcal{A}(u), v, \mathcal{A}(v))$, *then*

$$score(\mathcal{A}^*) \le \frac{2r(1+\alpha)}{1-\alpha n} score(\mathcal{A}^{**}),$$

where

$$r = \sup_{t_1,t_2\in T, s_1,s_2,s_2'\in S} \frac{p(t_1, s_1, t_2, s_2)}{p(t_1, s_1, t_2, s_2')} \ge 1.$$

Proof. Define

$$p(T', \mathcal{A}', T'', \mathcal{A}'') = \sum_{t'\in T'} \sum_{t''\in T''} p(t', \mathcal{A}'(t'), t'', \mathcal{A}''(t''))$$

then

$$\begin{aligned}
& score(\mathcal{A}^*) - score(\mathcal{A}_i) \\
= \;& m(T_i, \mathcal{A}^*) + p(T_i, \mathcal{A}*, T_i, \mathcal{A}*) + p(T_i, \mathcal{A}*, T - T_i, \mathcal{A}*) + p(T - T_i, \mathcal{A}*, T_i, \mathcal{A}*) - m(T_i, \mathcal{A}**) \\
& -p(T_i, \mathcal{A}**, T_i, \mathcal{A}**) - p(T_i, \mathcal{A}**, T - T_i, \mathcal{A}*) - p(T - T_i, \mathcal{A}*, T_i, \mathcal{A}**) \\
= \;& m(T_i, \mathcal{A}*) + p(T_i, \mathcal{A}*, T_i, \mathcal{A}*) + 2p(T_i, \mathcal{A}*, T - T_i, \mathcal{A}*) \\
& -m(T_i, \mathcal{A}**) - p(T_i, \mathcal{A}**, T_i, \mathcal{A}**) - 2p(T_i, \mathcal{A}**, T - T_i, \mathcal{A}*).
\end{aligned}$$

$$\begin{aligned}
& score(\mathcal{A}*) - score(\mathcal{A}i) - \alpha score(\mathcal{A}i) \\
= \;& m(T_i, \mathcal{A}*) + p(T_i, \mathcal{A}*, T_i, \mathcal{A}*) + 2p(T_i, \mathcal{A}*, T - T_i, \mathcal{A}*) - (1+\alpha)[m(T_i, \mathcal{A}**) + p(T_i, \mathcal{A}**, T_i, \mathcal{A}**) \\
& +2p(T_i, \mathcal{A}**, T - T_i, \mathcal{A}*)] - \alpha[m(T - T_i, \mathcal{A}*) + p(T - T_i, \mathcal{A}*, T - T_i, \mathcal{A}*)] \le 0
\end{aligned}$$

Move positive items to the left side and negative items to the right side, and sum up with $i = 0, 1, 2, \cdots, n$, we have, the left side

$$\begin{aligned}
L &= \sum_{i=0}^{n} \Big(m(T_i, \mathcal{A}*) + p(T_i, \mathcal{A}*, T_i, \mathcal{A}*) + 2p(T_i, \mathcal{A}*, T - T_i, \mathcal{A}*) \Big) \\
&\ge \sum_{i=0}^{n} \Big(m(T_i, \mathcal{A}*) + p(T_i, \mathcal{A}*, T_i, \mathcal{A}*) + p(T_i, \mathcal{A}*, T - T_i, \mathcal{A}*) \Big) \\
&= score(\mathcal{A}*)
\end{aligned}$$

the right side

$$\begin{aligned}
R &= \sum_{i=0}^{n} \Big((1+\alpha)[m(T_i, \mathcal{A}**) + p(T_i, \mathcal{A}**, T_i, \mathcal{A}**) + 2p(T_i, \mathcal{A}**, T - T_i, \mathcal{A}*)] \\
&\quad +\alpha[m(T - T_i, \mathcal{A}*) + p(T - T_i, \mathcal{A}*, T - T_i, \mathcal{A}*)] \Big) \\
&\le \sum_{i=0}^{n} \Big((1+\alpha)[m(T_i, \mathcal{A}**) + p(T_i, \mathcal{A}**, T_i, \mathcal{A}**) + 2r * p(T_i, \mathcal{A}**, T - T_i, \mathcal{A}**)] \\
&\quad +\alpha[m(T - T_i, \mathcal{A}*) + p(T - T_i, \mathcal{A}*, T - T_i, \mathcal{A}*)] \Big) \\
&\le 2r(1+\alpha) score(\mathcal{A}**) + \alpha n * score(\mathcal{A}*)
\end{aligned}$$

By adjusting the inequality of $L \le R$, we can obtain the conclusion. □