A Branch-and-Bound Approach to Knowledge-based Protein Structure Assembly

Gaolin Zheng and Giri Narasimhan

Abstract—**With the unprecedented growth in the size of sequence and structure databases, knowledge-based methods have become increasingly feasible for protein structure prediction. We developed a branch-and-bound method for structlets-based protein structure assembly. We explore the effectiveness of this approach by examining its capability to reconstruct the 3D structure of some proteins with known 3D structures. Although our algorithm involves exhaustive search, our BestFirst implementation of a branch-and bound strategy is able to eliminate around 2/3 of the total search space in order to find the optimal 3D assembly for a protein of interest.**

I. Introduction

The prediction of the three-dimensional structure of a protein, when only the amino acid sequence is known, has been a problem of considerable interest for many years. During the past several years, several entire genomes have been sequenced, ranging from those of short prokaryotes to the three billion base pair human genome. The genome projects generate huge amounts of biological sequence data which include sequence of complete genomes, sequence of complete sets of proteins (proteomes) [1]. It is reported that using only sequence information can help to assign function to only about 17% of all protein sequences in complete genomes [2]. In contrast, exploiting structural information to the largest possible extent could yield assignments of function to up to 50% of the proteins [3]. Currently, the rate of new protein sequences is growing exponentially with respect to the rate of protein structures being solved by experimental methods such as x-ray diffraction and nuclear magnetic resonance (NMR). It is a daunting task to determine the 3D structures of all sequenced proteins. In many cases, even a crude or approximate model can significantly help an experimentalist in guiding his/her experiments. The role of protein structure prediction is to predict unknown protein structures approximately and efficiently, so that we can roughly assign biological functions to the proteins [4].

Approaches to predict protein structure have ranged from purely *ab initio* methods [5] that are based on physical and chemical properties, to knowledge-based methodologies, such as homology modeling [6-8] or threading methods [9, 10], which depend on the presence of sequentially or structurally homologous proteins in the databases.

Knowledge-based protein structure prediction is becoming increasingly significant with the fast expansion of sequence and structure databases. The strategy is to use the knowledgebase to guess possible substructures (structlets) for subsequences of the given protein sequences and to then assemble it into a complete 3D protein structure. The problem is that subsequences can have several alternate substructures (structlets) in the knowledgebase. In this paper, we describe a branch-and-bound approach for knowledge-based protein structure assembly from the structlets.

II. METHODS

First, we would like to include some important terms used in this method.

***Seqlet*:** A sequence pattern appearing frequently (two or more times) in a given sequence database. It is a string of the form $(\Sigma\cup\{[\Sigma\Sigma^*\Sigma])(\Sigma\cup\{\text{`.'}\}\cup\{[\Sigma\Sigma^*\Sigma]\})^*(\Sigma\cup\{[\Sigma\Sigma^*\Sigma]\})$, where Σ={A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}. Strings such as M..E.A.P.[AD].L and R.A.. L[LR]AADM.F..E..I..GK are examples of seqlets. We assume that the average length of seqlets is about 20 [11].

***Structlet*:** A sequence of 3D coordinates retrieved from PDB, which matches a seqlet.

***1D biodictionary*:** A collection of frequently occurring amino acid combinations, referred to as *seqlets*.

***3D biodictionary*:** A collection of seqlets and their corresponding 3D structlets.

Seqlets can be produced using pattern discovery techniques such as the one used in TEIRESIAS [12]. Seqlets provide a comprehensive finite set of descriptors for protein sequence space. A 3D biodictionary is obtained by intersecting 1D biodictionary with a structural database such as protein data bank (PDB). Fig.1 shows the major steps in this approach.

Once seqlets and corresponding structlets are generated from the target protein, we need to assemble these structlets in a way that gives the minimum total RMSD (root mean square distance) in the regions of overlapping seqlets. Suppose n seqlets are generated from the target protein, and the average number of structlets matching each seqlet is m, then we will have to search from a tree of height n with up to m^n total nodes. For a protein of medium size (~500 residues), we tend to get about 50 overlapping seqlets. If the

Manuscript received July 2, 2008.

G. Zheng is with the Department of Mathematics and Computer Science, North Carolina Central University, Durham, NC 27707 USA (phone: 919-530-5110; e-mail: gzheng@nccu.edu).

G. Narasimhan is with School of Computing and Information Sciences, Miami, FL 33199 USA (e-mail: giri@cs. fiu.edu).

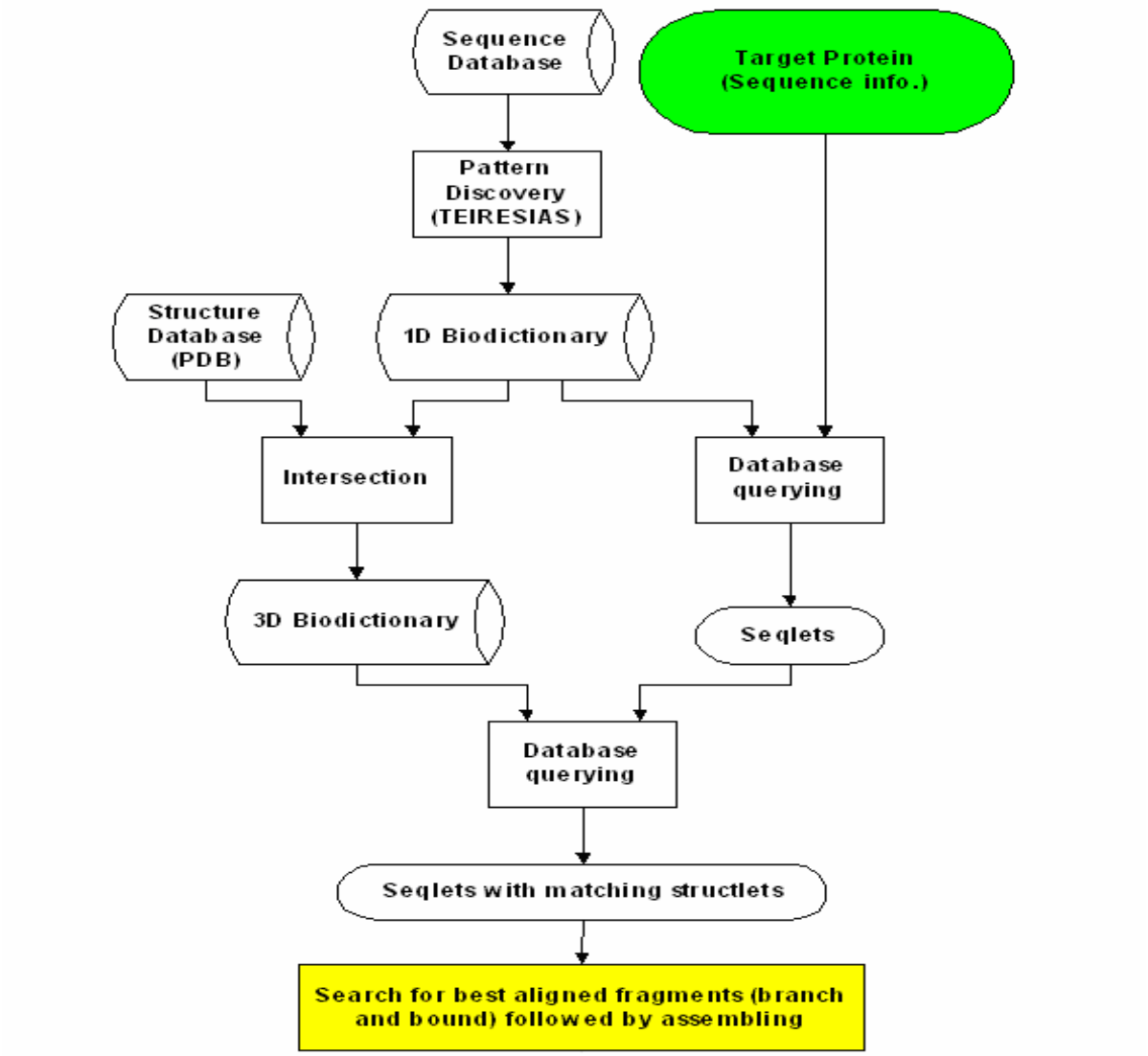

Fig. 1. The process of structlets-based protein structure assembly.

average number of matching structlets is 5, we will have to find an optimal solution from a search space of 5^{50} nodes. Every time a node is visited, an RMSD calculation is required. Computing the RMSD between two 3D structures is adapted from Schonemann's solution for orthogonal Procrustes problem [13]. The details of finding RMSD between two 3D structures A and B are shown in Fig. 2.

```
Input: Matrix A, B
Output: Root mean square distance between A and B
Procedure RMSD(A, B)
begin
  Move mass center of A to origin
  Move mass center of B to origin
  C := B'A
  Compute the SVD: [U, S, V]:= svd(C)
  Q := U*V'
  ||A-BQ||^2:=trace(A'A)+trace(B'B)-trace(Q'B'A)
  RMSD:= SQRT(||A-BQ||^2/N)
  Output RMSD
end
```

Fig. 2. RMSD computation of two 3D structures.

Since computation of RMSD of two structures is quite intensive, an exhaustive search without any constraints is prohibitively impossible, we adopted the concept of branch-and-bound and incorporated it into our searching strategy.

Branch-and-bound algorithms solve discrete optimization problems by examining the space of all combinatorial solutions (Branch), while using heuristics to eliminate possibilities that cannot lead to an optimal solution (Bound). In our case, optimization is a minimization process where solving the problem means finding a feasible solution with minimum cost. The total cost is computed as the total RMSD value of all the overlapping structlets in the potential solution chosen so far. A minimum total cost found so far is called global upper bound in our algorithm. Suppose we are currently visiting an internal node with a cost larger than the global upper bound, it is not necessary to visit the nodes branching from the current node based on elimination rule. The global upper bound is updated whenever a better solution is found. Details of the algorithm are given in Fig.3.

```
Input: a sequence of SN seqlets with matching structlets
Output: an optimal assembly of SN structlets to assemble the protein of
interest

Initialization:
minScore := infinity
score := infinity
choices := ∅
result := ∅
sort seqlets based on their location and length

procedure assemble(seqlets[SN])
begin
  N := number of structlets for the first seqlet
  for a := 1 to N do
     visit(seqlets[1].fragments[a], 1, 1)
  end
  combine the set of structlets from the result set
end

procedure visit(seqlets[i].fragments[j], i, k)
begin
  Compute score for node seqlets[i].fragments[j]
  Update the ith choice in the list of chosen fragments
  if i > SN then
    if score<minScore then
      minScore := seqlets[i].fragments[j].score
      result = choices[]
      return
   end
   return
  end
  else
    N := number of structlets for the ith seqlet
    for a := 1 to N do
      Compute RMSD value for node seqlets[i+1].fragments[a]
      if (seqlets[i+1].fragments[a].score < minScore) then
         visit(seqlets[i+1].fragments[a], i+1, k+1)
      end
    end
  end
end
```

Fig. 3. The branch-and-bound algorithm for structlets-based protein structure assembly.

We implemented both the naïve version and a BestFirst version for the above algorithm. In the naïve version, branches are visited sequentially in the order in which they are encountered. However, in the BestFirst version, the branches are sorted based on the child's alignability with its parent node. The branches are visited in the ascending order of their RMSD values of alignment with its parent node. The method is justified with the hope that a global upper bound will be reached earlier and more nodes will be eliminated during the searching. Therefore, it will make the searching faster and more efficient.

III. SIMULATION EXPERIMENTS

Currently this work is still in the prototype stage. We evaluate its effectiveness through simulated data sets. To construct simulated data, the 3D coordinates of alpha carbon atoms of a chosen protein are first extracted from Protein Data Bank data files. To simulate seqlets, overlapping amino acid sequences are randomly generated. The lengths of seqlets are chosen uniformly at

random from the range 5 through 35. The subsequences were generated to cover all the sequences in the protein. For each subsequence generated, we randomly (either from uniform distribution, or normal distribution or a combination) perturbed the 3D coordinates of the original protein to generate structlets at different degrees of perturbation. Then the fragments were randomly rotated (rotation angles along X, Y and Z axes were generated uniformly at random) to simulate the real 3D biodictionary searching results. Simulation data was also generated for proteins of varying lengths. A preliminary study shows that BestFirst implementation is up to 30 times faster than its naïve counterpart. Therefore, the BestFirst algorithm was used in the main experiments.

We ran our experiments on the proteins 1rhd, 3hsc, 2cro and 2yhx. We first wanted to know if our algorithm is able to recover the 3D structures for the proteins of interest. We also wanted to know how our algorithm scales as the number of seqlets increases. We chose a series of values for the number of seqlets ranging from 4 to 64. Due to the random nature of the simulation experiments, seven experiments were conducted for each assembly job. The total running time and the average depth of the nodes were recorded for each experiment.

IV. RESULTS AND CONCLUSION

From simulated data sets, our algorithm is able to recover the 3D structures for the four experimented proteins (Table 1).

Table 1. 3D protein structure assembly simulation results through branch-and-bound. (RMSD: root mean square distance between original structure and assembled structure).

PDB ID	# of residues	RMSD (Å)
1rhd	293	0.2577
3hsc	386	0.3436
2cro	69	0.0814
2yhx	457	0.7816

In general, we were able to reassemble the structures of the four experimented proteins in an acceptable manner. The deviation from original structure is less for smaller proteins and more for larger proteins.

The total running time for assembling different number of seqlets is shown in Fig. 4.

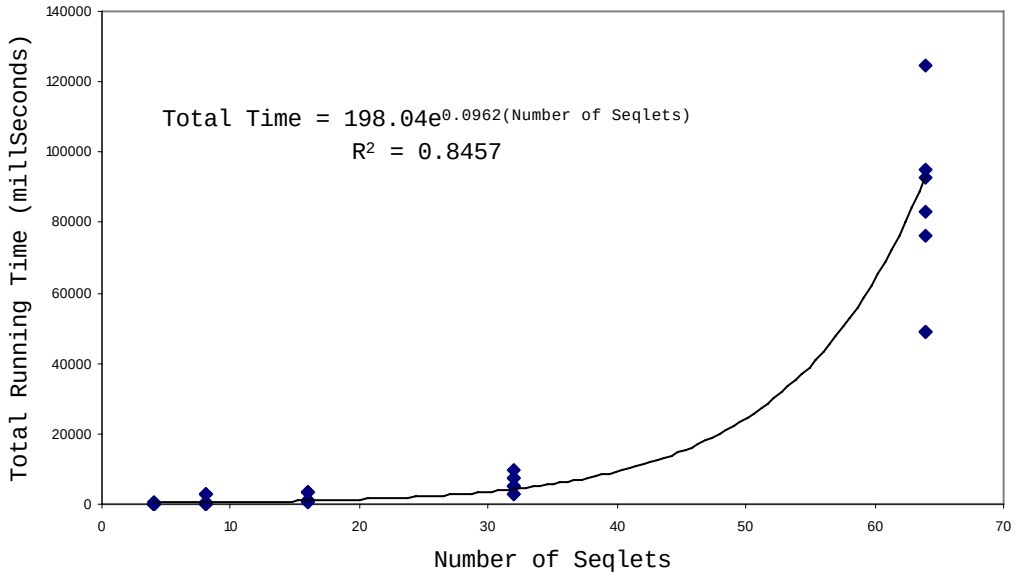

Fig. 4. Regression plot and functions of total running time versus the number of seqlets to be assembled

As seen in Fig. 4, the total running time increases exponentially with respect to the number of seqlets to be assembled. The average depth of nodes visited is shown in Fig. 5.

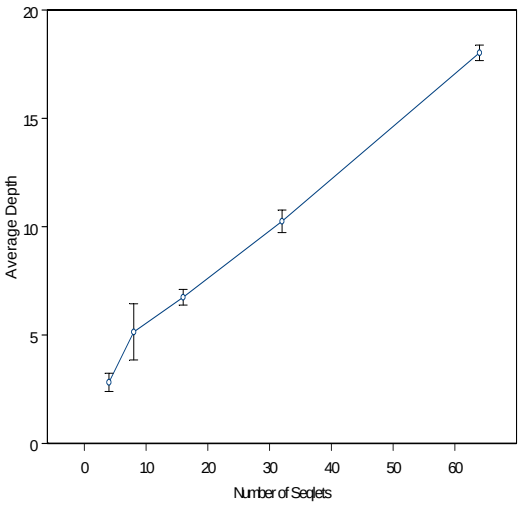

Fig. 5. The average depth of nodes visited versus the number of seqlets to be assembled.

The average depth of nodes visited was generally less than 1/3 of the height of the tree which is the number of seqlets (Fig.5). It is especially the case for larger number of seqlets. This indicates that our BestFirst implementation of the algorithm generally visited at most a cube root of the total number of nodes in the search tree.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we introduced a branch-and-bound algorithm to perform structlets-based protein structure assembly. Based on the simulation results, our method is by and large capable of assembling the structure fragments (structlets) into corresponding original protein structures. However, the computation is intensive and our current implement is generally feasible for small scale protein structure assembly. For large scale protein structure assembly, a parallel implementation is needed. Since the assembly problem is essentially embarrassingly parallel, a Master-Slave paradigm will be an obvious choice. A hierarchical Master-Slave paradigm might be necessary for massively parallel processors. Currently the program is implemented in Java and available upon request. Our future work will involve rewriting the source code in C and MPI and porting the application to cluster environments and supercomputers.

REFERENCES

[1] Sunyaev, S., W.r. Lathe, and P. Bork, Integration of genome data and protein structures: prediction of protein folds, protein interactions and "molecular phenotypes" of single nucleotide polymorphisms. Current Opinion in Structural Biology, 2001. 11(1): p. 125-130.

[2] Andrade, M.A., et al., Automated genome sequence analysis and annotation. Bioinformatics (Oxford, England), 1999. 15(5): p. 391-412.

[3] Koppensteiner, W.A., et al., Characterization of novel proteins based on known protein structures. Journal of Molecular Biology, 2000. 296(4): p. 1139-1152.

[4] Domingues, F.S., W.A. Koppensteiner, and M.J. Sippl, The role of protein structure in genomics. FEBS Letters, 2000. 476(1-2): p. 98-102.

[5] Osguthorpe, D.J., Ab initio protein folding. Current Opinion in Structural Biology, 2000. 10(2): p. 146-152.

[6] Warme, P.K., et al., Computation of structures of homologous proteins. Alpha-lactalbumin from lysozyme. Biochemistry, 1974. 13(4): p. 768-782.

[7] Clark, D.A., J. Shirazi, and C.J. Rawlings, Protein topology prediction through constraint-based search and the evaluation of topological folding rules. Protein Engineering, 1991. 4(7): p. 751-760.

[8] Fischer, D., et al., Assigning amino acid sequences to 3-dimensional protein folds. The FASEB Journal: Official Publication of the Federation of American Societies for Experimental Biology, 1996. 10(1): p. 126-136.

[9] Jones, T.A. and S. Thirup, Using known substructures in protein model building and crystallography. The EMBO Journal, 1986. 5(4): p. 819-822.

[10] Johnson, M.S., J.P. Overington, and T.L. Blundell, Alignment and searching for common protein folds using a data bank of structural templates. Journal of Molecular Biology, 1993. 231(3): p. 735-752.

[11] Rigoutsos, I., et al. Building Dictionaries Of 1D and 3D Motifs by Mining the Unaligned 1D Sequence of 17 Archaeal and Bacterial Genomes. in Proceedings Seventh International Conference on Intelligent Systems for Molecular Biology (ISMB '99). 1999. Heidelberg, Germany.

[12] Rigoutsos, I. and A. Floratos, Combinatorial Pattern Discovery in Biological Sequences: the TEIRESIAS Algorithm. Bioinformatics, 1998. 14(1): p. 55-67.

[13] Schonemann, P., A generalized solution of the orthogonal procrustes problem. Psychometrika, 1966. 31: p. 1-10.