An Efficient Branch-and-Bound Algorithm for the Assignment of Protein Backbone NMR Peaks

Guohui Lin
Department of Computing Science, University of Alberta
Edmonton, Alberta T6G 2E8, Canada. Email: ghlin@cs.ualberta.ca

Dong Xu
Protein Informatics Group, Life Sciences Division, Oak Ridge National Laboratory
Oak Ridge, TN 37831-6480, USA. Email: xud@ornl.gov

Zhi-Zhong Chen
Department of Mathematical Sciences, Tokyo Denki University
Hatoyama, Saitama 350-0394, Japan. Email: chen@r.dendai.ac.jp

Tao Jiang
Department of Computer Science, University of California, Riverside
Riverside, CA 92521, USA. Email: jiang@cs.ucr.edu

Jianjun Wen
Department of Computer Science, University of California, Riverside
CA 92521, USA. Email: wjianju@cs.ucr.edu

Ying Xu
Protein Informatics Group, Life Sciences Division, Oak Ridge National Laboratory.
Oak Ridge, TN 37831-6480, USA. Email: xyn@ornl.gov

Abstract

NMR resonance assignment is one of the key steps in solving an NMR protein structure. The assignment process links resonance peaks to individual residues of the target protein sequence, providing the prerequisite for establishing intra- and inter-residue spatial relationships between atoms. The assignment process is tedious and time-consuming, which could take many weeks. Though there exist a number of computer programs to assist the assignment process, many NMR labs are still doing the assignments manually to ensure quality. This paper presents a new computational method based on our recent work towards automating the assignment process, particularly the process of backbone resonance peak assignment. We formulate the assignment problem as a constrained weighted bipartite matching problem. While the problem, in the most general situation, is NP-hard, we present an efficient solution based on a branch-and-bound algorithm with effective bounding techniques and a greedy filtering algorithm for reducing the search space. Our experimental results on 70 instances of (pseudo) real NMR data derived from 14 proteins demonstrate that the new solution runs much faster than a recently introduced (exhaustive) two-layer algorithm and recovers more correct peak assignments than the two-layer algorithm.

Keywords: NMR, chemical shift, peak assignment, protein structure, bipartite matching, "greedy" algorithm, branch-and-bound, combinatorial technique.

1 Introduction

A major objective of bioinformatics is to develop efficient algorithms for technologies that generate high-throughput biological data. Due to the efforts of structural genomics [17], the NMR (nuclear magnetic resonance) technique has been used as a high-throughput technology to quickly identify protein structures at a genome scale. However, protein structure determination by NMR is still a tedious process, compared with other high-throughput technologies such as DNA sequencing and microarray. Typically NMR protein structure determination involves the following steps: (i) NMR data generation — producing resonance peaks in 1-, 2-, 3- and 4-D spaces[1], which may reflect both "signature" information of amino acid types and connectivity (adjacency) information between amino acids; (ii) peak picking — identifying "real" resonance peaks (peaks generated from protein atoms rather than noise) from NMR spectral maps; (iii) peak assignment — assigning resonance peaks, typically peak groups, to individual residues of the target protein sequence; (iv) structural restraint extraction — extracting inter-residue distances, dihedral angles, etc., based on the peak assignment; and (v) structure calculation — calculating the protein structure, using molecular simulation and energy minimization, under the identified NMR restraints. Thanks to advances in NMR technology, running NMR experiments and collecting necessary data for structure determination are becoming easier and easier. The bottleneck for NMR structure determination is often in the stage of data analysis.

Peak assignment is a very time-consuming process in an NMR structure determination, and usually it takes weeks of manual work. The peak assignment process is typically done in an iterative fashion through establishing relationships between different NMR spectra [16]. It involves (a) mapping peaks from different NMR spectra to the same residues, where the peaks originated from the same residue are represented by a *spin system*; (b) identifying the possible amino acid types of each spin system; (c) identifying adjacency relationships (neighboring residues in the protein sequence) between spin systems and linking them into contiguous segments; and (d) assigning the segments and the remaining isolated spin systems to the residues of the protein sequence. Two key pieces of information used in the assignment process are (1) different amino acid types have different distributions of spin systems, and (2) adjacency information (in the protein sequence) between spin systems obtained by identifying their common resonance frequencies. A great deal of work has been done [21, 5, 6, 22, 19, 3, 15, 25, 12], with the aim to automate the peak assignment process.

One of the most useful piece of information in a manual peak assignment process is the connectivity information among adjacent spin systems, which was used in the above formulation. Existing procedures [21, 7] can be used to determine which spin systems are connected in the protein sequence, with some level of accuracy. The result of such procedures is a set of (short) segments of connected spin systems, which correspond to some protein sequence segments. The basic idea of such a procedure is that there are inter-residual peaks in multidimensional NMR spectra that convey the connectivity information between residues. For example, in HNCA spectrum, an inter-residual peak $({}^{1}\mathrm{H}_i^N, {}^{15}\mathrm{N}_i, {}^{13}\mathrm{C}_{\alpha i-1})$ signifies the sequential adjacency relationship between two intra-residual peaks (spin systems), $({}^{1}\mathrm{H}_{i-1}^N, {}^{15}\mathrm{N}_{i-1}, {}^{13}\mathrm{C}_{\alpha i-1})$ and $({}^{1}\mathrm{H}_i^N, {}^{15}\mathrm{N}_i, {}^{13}\mathrm{C}_{\alpha i})$. By identifying each intra- and inter-residual peak and then correlating them by using their chemical shifts, it is straightforward to obtain the connectivity information. In this work, we have used a pair of HNCA and HN(CO)CA experiments to infer segments of connected spin systems. We will focus on the peak assignment of backbone atoms. This is the initial step for peak assignments, and it is particularly useful for quickly determining a low-resolution structure of protein backbones.

The backbone peak assignment problem with connectivity information has recently been formulated as a constrained weighted bipartite matching problem on two disjoint groups, one group containing spin systems and the other containing a sequence of amino acids with predicted secondary structures [24]. A weighted bipartite matching problem [14] is to find a one-to-one matching between elements of two groups that maximizes the total weight, where each matched pair of elements has a pre-specified weight. NMR peak assignment can be naturally modeled as a weighted bipartite matching problem, where each weighted edge represents a possible assignment of a spin system to an amino acid on the protein sequence with a quantitative confidence value. To incorporate the connectivity information, we can extend the above model to a *constrained* weighted bipartite matching problem, *i.e.* we define a (partial) neighboring relationship between the elements of each group and require that neighbors of a group be matched only with neighbors of the other group. Unfortunately, the constrained bipartite matching problem is NP-hard, even if the edges are unweighted [24]. Some heuristics have been proposed very recently, including a slow, exhaustive two-layer algorithm [24] and some fast approximation algorithms [8]. These heuristics attempt to find feasible matchings with (approximately) the largest weights, and may work well for the NMR peak assignment based on our observation that, in practice, a segment of connected spin systems typically have a better score at the "correct" assignment position (*i.e.* the matching between a spin system of NMR peaks and the residue that generates the peaks) than almost all other (incorrect) assignment positions, especially as the size of the segment increases. Among these heuristics, the two-layer algorithm performs the best overall in terms of recovering correct assignments. The idea of the two-layer algorithm is to filter out most of the impossible assignments for connected spin systems, and then

[1]The n-D space means a measurement of the chemical shifts of n types of atoms in a protein.

carry out an exhaustive search for all combinations of the assignments for these connected spin systems to obtain a global optimal assignment. The method works well in general and is able to recover most of the correct assignments in the our tests, especially when the connectivity information is “dense”. However, it sometimes produces poor assignments because the correct assignments for some connected spin systems are filtered out in the first layer. It may also take an extremely long time because of the exhaustive nature of its second layer, when the filtering in its first layer is not very effective. Hence, better algorithms for constrained bipartite matching would be desirable and they may lead to practical solutions for the real NMR peak assignment problem.

To overcome the drawbacks in the two-layer algorithm, we develop a method for finding maximum-weight constrained bipartite matchings based on the branch-and-bound technique and a “greedy” filtering algorithm. The branch-and-bound algorithm uses an efficient (unconstrained) bipartite matching algorithm and the approximation algorithms in [8] to compute necessary lower bound on the solution to help prune the search tree, and returns an optimal solution, *i.e.* a feasible matching with the largest weight. It runs much faster than the exhaustive search used in the two-layer algorithm. Although the branch-and-bound algorithm works well in terms of assignment accuracy, we introduce a greedy filtering algorithm to accelerate the computation substantially without affecting the assignment accuracy significantly. The greedy algorithm divides a constrained bipartite matching problem for the whole protein into several smaller constrained bipartite matching problems based on some preliminary assignments, and calls the branch-and-bound algorithm to solve each of these problems. Although the method (*i.e.* branch-and-bound augmented with greedy filtering) may potentially take exponential time in the worst case, our experiments on real and simulated NMR data illustrate that it runs reasonably fast (much faster than the two-layer algorithm) when the spin system connectivity is sufficiently “dense”. In other words, the bounding and filtering techniques become very effective when the graph contains many large segments of connected spin systems. In particular, the test results on 70 instances of NMR peak assignment data derived from 14 proteins, each being associated with 5 different degrees of connectivity, show that our new algorithm runs much faster than the two-layer algorithm and recover more correct peak assignments than the two-layer algorithm in most of the cases.

The paper is organized as follows. Section 2 discusses the method of scoring a matching between spin systems and amino acids, which is crucial for the formulation of NMR peak assignment as a constrained weighted bipartite matching problem. It also explains the source of real NMR spectral data that can be used to test the effectiveness of peak assignment algorithms. Section 3 describes the constrained weighted bipartite matching problem, and presents the the branch-and-bound algorithm for the constrained weighted bipartite matching problem and the greedy filtering algorithm. Section 4 shows the experimental results on the performance of the branch-and-bound algorithm augmented with greedy filtering. Section 5 concludes the paper with some suggestions of future research directions.

2 Scoring Scheme and Data Source

NMR resonance of an atom is measured by its *chemical shift*, a characteristic value of the atom type and its physi-chemical environment. A combination of chemical shifts of different atom types of an amino acid, e.g., $^1\mathrm{H}_\alpha$, $^{13}\mathrm{C}_\alpha$, $^1\mathrm{H}^N$, and $^{15}\mathrm{N}$, provides highly useful information for distinguishing different types of amino acids. It is possible to predict the chemical shifts of each amino acid if we know a protein’s 3-D structure. However, the prediction becomes much more challenging when the structural information is not available. So far, chemical shift prediction has only been successful for proteins which have close homologs with known chemical shifts [11] or solved high-resolution structures [20].

Instead of attempting to predict the chemical shifts of a protein, we have developed a statistics-based scoring scheme for assessing the likelihood of an array of particular chemical shifts that may be produced by a particular amino acid type. To account for some structural information, we include the predicted secondary structures in our scoring scheme. We have used the PSIPRED program [13] for secondary structure predictions, which is one of the best secondary structure prediction programs available with about 80% accuracy for assigning a residue to an α-helix, β-strand, or a loop. In the current study, we consider only the chemical shifts of $^{13}\mathrm{C}_\alpha$ and $^1\mathrm{H}_\alpha$ atoms, without using $^1\mathrm{H}^N$ and $^{15}\mathrm{N}$’s chemical shifts as we found that they are not particularly useful for backbone peak assignment.

The basic idea of our scoring scheme is to examine a set of solved NMR structures with available chemical shifts in the BioMagResBank database [18], and collect the following statistics: the relative frequency of a particular triplet (aa, ss, cs), for each amino acid type aa, each secondary structure type ss, and each chemical shift range of cs (originated from $^{13}\mathrm{C}_\alpha$ or $^1\mathrm{H}_\alpha$). Then we estimate the *expected* frequency of each triplet. The ratio of the two frequencies provides an estimate of the preference of an chemical shift coming from a particular amino acid type sitting inside a particular secondary structure. More specifically, we have divided all chemical shifts of C_α and H_α into five bins such that each bin is approximately equally populated using 220 proteins from the BioMagResBank database. Hence, cs will be of a discrete value in the range of 1 through 5. For each (aa, ss, cs), we calculate two scores $score(aa, ss, cs|H_\alpha)$ and $score(aa, ss, cs|C_\alpha)$; and use $score(aa, ss, cs|H_\alpha) + score(aa, ss, cs|C_\alpha)$ as the weight of each edge (aa, cs) (actually this number multiplied by -1000). Further details of calculating the edge weights can be found in [24]. Four types of NMR spectra data are used in our work: (1) 2D (^{1}H,^{15}N)-HSQC, (2) HNCA, (3) H(CA)NH, and (4) HN(CO)CA.

To test our new algorithm in this work, we selected 14 proteins without solved atomic structures, obtained their chemical shifts of ^{13}C$_\alpha$ and ^{1}H$_\alpha$ from BioMagResBank [18], and created connectivities via simulation. Since these proteins do not have solved atomic structures, none of them is in the data set that was used to derive the scoring function. Thus, test results on these proteins will be unbiased with respect to the score function.

3 Algorithms

We first formulate the peak assignment problem as a weighted bipartite matching problem. A *bipartite graph* is a graph $G = (N, E)$ whose vertex set N can be partitioned into two disjoint sets U and V such that each edge of E has one vertex in U and one vertex in V. We consider only undirected bipartite graphs, *i.e.* edges of E have no orientations. A *matching* of G is a set of edges of E, no two of which share a vertex of N. A *perfect* matching is a matching that covers all vertices of the graph. In a *weighted* bipartite graph, each edge $e \in E$ has a weight $w(e)$. The *maximum weight (perfect) matching problem* of G is to find a (perfect) matching with the highest total weight possible. The maximum weight matching problem provides a natural formulation for the peak assignment problem. Let $U = u_1u_2...u_n$ represent a protein sequence with n residues, and $V = \{v_1, v_2, ..., v_m\}$ be its observed spin systems. In our application, each v_i is a 2D vector of chemical shifts of ^{1}H$_\alpha$ and ^{13}C$_\alpha$. Ideally, each u_i of U has one spin system, corresponding to exactly one v_j. Due to the non-trivial nature of resonance peak picking [4], some of the actual peaks may not be picked (missing from the V list) and some of the peaks in V could be noise. A bipartite graph representation of the peak assignment problem can be defined as follows. Every residue u_i is made a vertex of set U and $V = \{v_1, v_2, ..., v_m\}$ as given. Each pair of vertices u_i and v_j are connected by an edge with the weight of $w(u_i, v_j)$ (the integer value of $-1000 \times score(aa, ss, cs)$). One could formulate the peak assignment problem as finding a maximum weight matching for the above bipartite graph. An efficient algorithm for finding a maximum weight bipartite matching, *e.g.* the implementation by Goldberg and Kennedy [10] using a technique called *push and relabel*, could be used for this purpose. However, our experience with this approach has shown that the resulting assignments are usually unsatisfactory because the connectivity is totally ignored. Thus, here we formulate the peak assignment problem with connectivity information as a constrained bipartite matching problem. Let $G = (U \cup V, E)$ be a bipartite graph, where each edge $(u_i, v_j) \in E$ has a weight $w(u_i, v_j)$. In addition, consecutive elements of $V = \{v_1, \ldots, v_m\}$ may form non-overlapping strings, e.g., $V = \{v_1, v_2v_3v_4, v_5v_6, v_7, \ldots, v_{m-3}v_{m-2}v_{m-1}v_m\}$. A constrained bipartite matching is a bipartite matching such that if $v_p \ldots v_q$ is a string of V and v_p is matched with u_i, then v_{p+1} should be matched with u_{i+1}, …, and v_q should be matched with $u_{i+(q-p)}$, *i.e.* a string of V should be matched with consecutive elements of U in order. The *constrained bipartite matching problem* is to find a constrained matching that has the maximum possible weight. The constrained bipartite matching problem is shown to be NP-hard and MAX SNP-hard [8, 24], even if all edges have unit weight and every string in V has length at most 2.

In this section, we will describe three algorithms: (1) two approximation algorithms introduced in [8] that will be used to estimate lower bounds in the branch-and-bound algorithm; (2) the branch-and-bound algorithm for assigning spin systems to residues on a protein sequence in the constrained bipartite matching problem. (3) the greedy filtering algorithm to speed up the branch-and-bound algorithm by reducing the search space.

3.1 Two efficient approximation algorithms

Following the notations above, we consider an instance of constrained weighted bipartite matching: $G = (U \cup V, E)$, where $U = \{u_1, u_2, \ldots, u_{n_1}\}$, $V = \{v_1 \cdots v_{i_1}, v_{i_1+1} \cdots v_{i_2}, \ldots, v_{i_p} \cdots v_{n_2}\}$, and $E \subseteq U \times V$ is the set of edges. Here, $v_{i_{j-1}+1} \cdots v_{i_j}$ in V denotes a string of consecutively adjacent spin systems. We may assume that for every substring v_jv_{j+1} of a string in V, $(u_i, v_j) \in E$ if and only if $(u_{i+1}, v_{j+1}) \in E$. Based on $G = (U \cup V, E)$, we construct a new edge-weighted bipartite graph $G' = (U \cup V, E')$ as follows: For each $u_i \in U$ and each string $v_jv_{j+1} \cdots v_k \in V$ such that $(u_i, v_j) \in E$, let (u_i, v_j) be a super-edge in E' and its weight be the total weight of edges $\{(u_{i+x}, v_{j+x}) \mid 0 \le x \le k - j\}$ in E.

We have recently developed two efficient approximation algorithms for computing a feasible matching for G' [8]. The idea behind the first approximation algorithm is as follows. Let M^* denote an optimal matching on G'. It can be easily shown that there is a vertex such that every (super-)edge incident to which conflicts with at most two edges in M^*. Using this fact and the *local ratio* technique in [2], we can construct a recursive algorithm to find a (good) feasible matching on G' with weight at least half of the optimum. (The algorithm in fact, as we were informed very recently, follows directly from the recent result of Bar-Noy *et al.* [1]

on interval scheduling.)

The key idea behind the second is that if the length of the longest string in V is at most four times the length of the shortest string, then a simple greedy algorithm always finds a constrained bipartite matching whose weight is at least one-sixth of the optimal. So, if we partition the strings of V into $\log_4 D$ groups, where D is the length of the longest string in V, and greedily find a constrained bipartite matching between each group and U, then the weight of the heaviest matching is at least $1/(6\log_4 D) = 1/(3\log_2 D)$ times the optimal. The algorithm runs in $O(|U||V|\log_2 D)$ time. Moreover, it outputs a matching with weight at least $1/(3\log D)$ times the optimal.

Although the first (2-approximation) algorithm has a better performance guarantee than the second ($3\log D$) algorithm in the worst case, our experimental results in [8] showed that the second algorithm in fact performs better generally on real NMR peak assignment data both in terms of maximizing the weight of the output matching and in terms of recovering a large number of correct matching edges. Because of the crude nature of some of the steps in the approximation algorithms, it is easy to see that these algorithms will fail to recover many correct matching edges when the strings in V are distributed in the worst cases. Nevertheless, they can be very effective in providing lower bounds in a branch-and-bound algorithm. Let W_3 denote the weight of the matching found by the $3\log D$-approximation algorithm and W_2 denote the weight of the matching found by the 2-approximation algorithm. Then we will use $\max\{W_2, W_3\}$ to lower bound the weight of M^* (the optimal feasible matching). Moreover, we will also use $2W_2$, in addition to the weight of the optimal (unconstrained) bipartite matching found by Goldberg and Kennedy algorithm, to upper bound the weight of M^*.

3.2 Branch-and-bound algorithm

Branch-and-bound is a systematic method for solving optimization problems. It is a general optimization technique that applies to many problems. Due to the inherent exponential complexity of many combinatorial problems, branch-and-bound may take exponential time in the worst case. On the other hand, if applied carefully, it can run reasonably fast on average for practical applications while not being able to overcome the worst case. The general idea of the branch and bound algorithm is to construct a search tree and apply a carefully selected criterion to determine which node to expand the search. Another criterion is to tell the algorithm when an optimal solution has been found. In this way, a substantial portion of the search space for the optimal solution can be avoided, in particular, when the lower bound for the maximum solution can be defined well (as done in our case).

The basic idea of applying the branch-and-bound algorithm in the assignment of protein backbone NMR peaks is to identify the lower bound of the total matching score and exclude any branch of possible matchings with scores smaller than the lower bound during the search for the matching with the globally maximum score. The scoring function has a physical meaning, *i.e.* a positive value indicates a favorable matching and a negative value indicates an unfavorable matching. For the correct peak assignment, the total score should be positive, although some pairs between spin systems and residues may contribute negative scores. Hence, 0 is a natural lower bound of the total matching score. If we find that a branch of possible matchings that satisfy the constraints have negative score, we can skip the whole branch for the purpose of searching the globally maximum score. As shown in Figure 1, possible matchings that satisfy the constraints predominantly have negative scores. That means a substantial portion of the search space can be ignored during the search. In fact, we can do even better than that. Our efficient approximation algorithms can define a positive lower bound that is very close to the globally maximum score. This will further exclude many possible matchings during the search. Figure 1 also suggests that the branch-and-bound algorithm may be more effective when the connectivity is dense, as a big portion of possible matchings have negative scores.

Here we show the mathematical formulation of our algorithm. A string is *long* if it contains at least three spin systems. Given $G' = (U \cup V, E')$, it is easy to notice that when a super-edge (u_i, v_j) is included into a matching and $v_j v_{j+1} \ldots v_k$ is the long string involved, then any other edges incident to vertices $u_i, u_{i+1}, \ldots, u_{i+k-j}$ should be removed from further consideration. Moreover, we can use this rule to remove the conflicting super-edges from further consideration as well. Assume that $v_j v_{j+1} \ldots v_k$ is the kth long string. Let $[l_k, r_k]$ denote the range of index i such that (u_i, v_j) is an edge of E'. For simplicity, we call $[l_k, r_k]$ the *matching range* of the kth long string. To build a branch-and-bound tree, set G' to be the root node. Compute approximately optimal matchings for the root by calling the 2-approximation and $3\log D$-approximation algorithms, and let W_0 be the maximum of the weights of the two output matchings. W_0 will be used as a global variable in the algorithm that keeps the best lower bound of the optimal matching(s).

Generally, for each branching node under consideration, if for every long string of index k, we have $l_k = r_k$, then the node is treated as a *leaf* node. In such a leaf node, every long string has to be assigned to its incident super-edge. If these super-edges agree with each other, we can delete all the super-edges from the graph and call Goldberg and Kennedy algorithm to compute a partial maximum weight matching for the remaining graph. The output partial matching com-

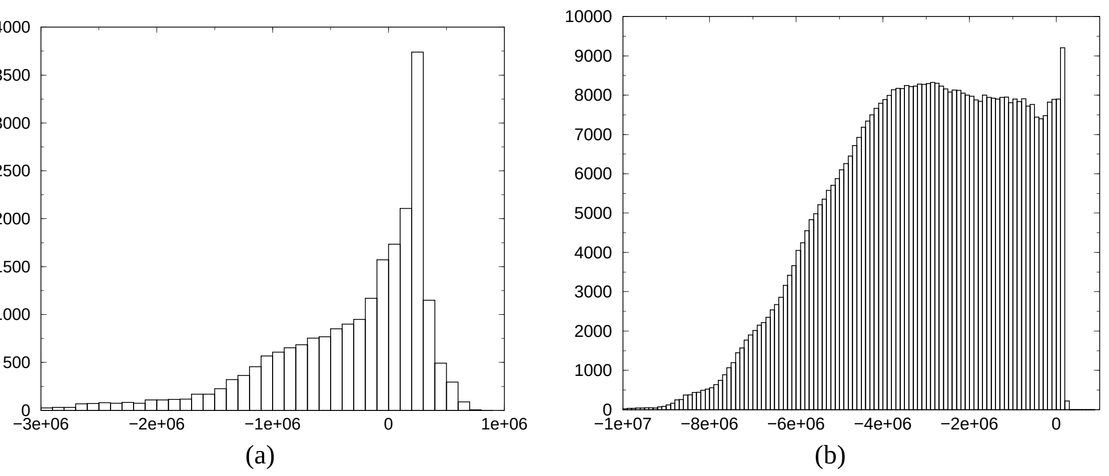

Figure 1. Distribution of scores for possible matchings that satisfy the constraints for protein 1avf. (a) All possible matchings when the number of adjacent pairs of spin systems is at 20% of the total number of residues. (b) A sampling of 500,000 possible matchings when the number of adjacent pairs of spin systems is at 80% of the total number of residues.

bined with the super-edges deleted earlier form a perfect matching of the instance. Note that this perfect matching is a feasible matching of G', the instance represented at the root. Therefore, if its weight is greater than W_0, then we update W_0 with this weight. On the other hand, if we detect that there is a pair of conflicting super-edges, then this leaf node is *killed*, since it does not yield any interesting matching.

Suppose that the branching node is not a leaf. That is, there is some long string with index k whose matching range satisfies $r_k > l_k$, or equivalently there are more than one super-edges incident to that long string. In this case, we compute approximately optimal matchings for the node instance by calling the 2-approximation and $3 \log D$-approximation algorithms. Let W_2 be the weight of the matching found by the 2-approximation algorithm. Since $2W_2$ is an upper bound of the optimal feasible matching weight for this branching node instance, if $2W_2 < W_0$ then we know that any feasible matching for this instance has its weight less than W_0 and thus would not be an optimal feasible matching for G'. In this case, the branching node is killed. A similar analysis can be done using Goldberg and Kennedy algorithm. Otherwise, if the larger of the weights of the above two approximately optimal matchings is greater than W_0, then we have just found a better feasible matching and we can update W_0 using this value. The branch-and-bound algorithm then searches or prunes the children of the node, which are defined below, in a standard way.

The children of a branching (non-leaf) node are defined as follows. Look for a longest range of the long strings, say $[l_k, r_k]$, and bipartition it to form two child nodes. In the left child node, the k-th long string has matching range $[l_k, r'_k]$, where $r'_k \leq (l_k + r_k)/2$ and is the largest index such that $(u_{r'_k}, v_j)$ is an edge in E'. Similarly, in the right child node, the k-th long string has matching range $[l'_k, r_k]$, where $l'_k > (l_k + r_k)/2$ and is the smallest index such that $(u_{l'_k}, v_j)$ is an edge in E'.

At the end, W_0 records the weight of an optimal feasible matching. Using a simple tracking technique, we can store the optimal matching itself as well. A pseudo-code for the branching and bounding steps at each node is given in Figure 2.

3.3 Augmenting branch-and-bound with greedy filtering

A major problem in the two-layer algorithm is its filtering procedure. The filtering procedure aims at removing improbable assignments of a string of constrained consecutive spin systems. Although in almost all cases the correct assignment of the string is not filtered out, an incorrect assignment can happen. An incorrect assignment of a string affects the assignments of other strings since different strings cannot overlap with each other in the solution. Hence, the cascade effect may result in very poor NMR peak assignment. To overcome this problem, we employ a greedy algorithm that takes the best immediate, or local solution for locating most probable assignments of all the long strings. As shown in [24], the correct assignment of a long string can stand out easily from the scoring function, compared with a short string. The greedy algorithm can

```
BRANCH-AND-BOUNDing at one node: current lower bound $W_0$
    if (no ranges are longer than 1)
        if (no pair of conflicting super-edges)
            remove super-edges and collect their total weight $W_1$;
            remove edges incident to the residues to which long strings match;
            call Goldberg and Kennedy algorithm to get a partial matching with weight $W_2$;
            if ($W = W_1 + W_2 > W_0$)
                $W_0 \leftarrow W$;
    else
        find a longest range, say $(l_k, r_k)$;
        branching to two child nodes:
            one with range $(l_k, r'_k)$ and the other with $(l'_k, r_k)$ for the $k$-th long string;
        for (each of the child node)
            call Goldberg and Kennedy algorithm to get an optimal unconstrained matching with weight $W_1$;
            if ($W_1 < W_0$)
                kill the node;
            else
                call 2-approximation algorithm to get an approximately optimal matching with weight $W_2$;
                if ($2W_2 < W_0$)
                    kill the node;
                else
                    call $3 \log D$-approximation algorithm to get another approximately optimal matching
                        with weight $W_3$;
                    if ($\max\{W_2, W_3\} > W_0$)
                        $W_0 \leftarrow \max\{W_2, W_3\}$
                    recursively call BRANCH-AND-BOUND at this child node;
```

Figure 2. The branching and bounding steps at a node.

provide a limited number of probable combinations for the assignments of long strings. For each combination, we can fix the long strings and solve the assignments of the short strings and unconnected spin systems using the branch-and-bound algorithm. In this way, we can save computing time substantially. The greedy algorithm may find suboptimal solutions, but we have observed in our experiments that it works much better than the filtering procedure used in the two-layer algorithm.

We now describe how the greedy algorithm works. Consider an instance of constrained weighted bipartite matching with a set of ℓ long strings $B = \{B_1, B_2, \ldots, B_\ell\}$, where string B_i with length j contains spin systems $\{v_{i_1}, v_{i_1+1}, \cdots, v_{i_1+j-1}\}$. Our greedy algorithm proceeds as follows for some pre-specified k:

1. Sort the ℓ long strings according to their lengths so that in $B = \{B_1, B_2, \ldots, B_\ell\}$, B_1 is the longest and B_ℓ is the shortest.

2. Find the best k assignments for B_1 on the protein sequence, *i.e.*, B_1^1, B_1^2, ..., B_1^k.

3. For each assignment B_1^j, $1 \leq i \leq k$, find the best k assignments for B_2. Out of the k^2 combinations of B_1 and B_2 assignments, select the top k combinations.

4. For each combination of B_1 and B_2 assignments, find the best k assignments for B_3. Out of the k^2 combinations of B_1, B_2, and B_3 assignments, select the top k combinations.

5. Repeat the above steps for $B_4, B_5, \ldots, B_\ell$.

After B_ℓ is considered, we will have obtained the best k assignments for all the long strings in B. These results can be combined with the branch-and-bound algorithm as follows. For each of above k assignments, we use the branch-and-bound algorithm to find an optimal assignment for the remaining short strings and isolated spin systems. Then we obtain the total score by adding the score of the long string assignment and the score of the assignment of the short strings and individual spin systems. We rank the above k overall assignments according to the total score, and use the best total score assignment as the final solution.

In our experiments, it was observed that $k = 10$ seems to give the best result.

	$\|M^*\|$	$w(M^*)$	$w(M_b)$	R_b	R_t		$\|M^*\|$	$w(M^*)$	$w(M_b)$	R_b	R_t
bmr4027_5	158	1896284	1934329	33	33	bmr4144_5	78	949170	997603	16	14
bmr4027_6			1921093	37	36	bmr4144_6			993361	11	30
bmr4027_7			1911828	74	79	bmr4144_7			954633	64	34
bmr4027_8			1894532	128	113	bmr4144_8			954585	67	67
bmr4027_9			1896606	156	155	bmr4144_9			952241	75	75
bmr4288_5	105	1249465	1255475	12	38	bmr4302_5	115	1298321	1331391	24	34
bmr4288_6			1261696	26	52	bmr4302_6			1324395	43	44
bmr4288_7			1251020	57	55	bmr4302_7			1323495	62	65
bmr4288_8			1238344	66	N/A	bmr4302_8			1308217	103	N/A
bmr4288_9			1249465	105	105	bmr4302_9			1298321	110	111
bmr4309_5	178	2048987	2117910	25	46	bmr4316_5	89	1029827	1009329	30	42
bmr4309_6			2110992	57	59	bmr4316_6			1022505	35	60
bmr4309_7			2093595	77	122	bmr4316_7			1029827	79	79
bmr4309_8			2067295	101	106	bmr4316_8			1029827	89	87
bmr4309_9			2048987	178	176	bmr4316_9			1029827	89	89
bmr4318_5	215	2390881	2497294	20	38	bmr4353_5	126	1498891	1532518	17	17
bmr4318_6			2481789	35	38	bmr4353_6			1524784	24	35
bmr4318_7			2444439	52	87	bmr4353_7			1516244	44	29
bmr4318_8			2420829	62	113	bmr4353_8			1472871	80	N/A
bmr4318_9			2393435	201	N/A	bmr4353_9			1498891	126	126
bmr4391_5	66	710914	753046	18	13	bmr4393_5	156	1850868	1874095	41	30
bmr4391_6			745501	10	10	bmr4393_6			1871616	59	45
bmr4391_7			735683	26	0	bmr4393_7			1862221	76	74
bmr4391_8			723111	42	18	bmr4393_8			1853749	130	128
bmr4391_9			710914	66	N/A	bmr4393_9			1851298	152	143
bmr4579_5	86	950173	967647	15	13	bmr4670_5	120	1391055	1435721	22	35
bmr4579_6			976720	32	15	bmr4670_6			1429449	30	48
bmr4579_7			958335	44	42	bmr4670_7			1402335	38	45
bmr4579_8			956115	63	49	bmr4670_8			1391055	116	N/A
bmr4579_9			950173	86	86	bmr4670_9			1391055	116	N/A
bmr4752_5	68	882755	884307	21	28	bmr4929_5	114	1477704	1496460	23	24
bmr4752_6			892520	32	28	bmr4929_6			1496954	32	24
bmr4752_7			887292	41	43	bmr4929_7			1490155	56	65
bmr4752_8			882755	68	N/A	bmr4929_8			1481593	88	86
bmr4752_9			882755	68	68	bmr4929_9			1477704	114	112

Table 1. Summary on the performances of the branch-and-bound (augmented with greedy filtering) and the two-layer algorithms on 70 instances of NMR peak assignment. The number after the underscore symbol in the name of each instance indicates the "density" of adjacency information in the instance (more precisely, _5 means that the number of adjacent pairs of spin systems is 50% of the total number of residues). We start from density index 5 because most graphs have this level of density in actual NMR experiments. M^* represents the correct assignment and, M_b (or M_t) is the assignment computed by the branch-and-bound (or the two-layer, respectively) algorithm. The parameters $R_b = |M^* \cap M_b|$ and $R_t = |M^* \cap M_t|$, show how many correct matching edges were recovered by the branch-and-bound and two-layer algorithms, respectively. Each N/A indicates the failure of a computation by two-layer algorithm. The weights shown in the table are the integer values of $-1000 \times score(aa, ss, cs)$).

4 Results

We have tested our new method described above (*i.e.* branch-and-bound augmented with greedy filtering) on 14 proteins, each with 5 cases of connectivity, as shown in Table 1. The computing time on a Pentium Ⅲ with 1GHz CPU and 1GB memory takes from less than 1 second to an hour for each case. There are 8 cases where the 2-layer algorithm failed since the algorithm was taking too long (on a supercomputer) and had to be killed before any result could be obtained. The new method succeeded on all these instances. The weight of its output matching is generally better than that obtained by the two-layer algorithm. It is in fact often better than the weight of the correct matching where every spin system is assigned correctly (see the Discussion Section for the reasons behind this). The number of correct matching edges recovered by the new algorithm is better than that of the two-layer algorithm in most of the cases. In some cases the improvement is very significant.

5 Discussion

In summary, we have developed an improved computational method for automated NMR peak assignment based on branch-and-bound and a greedy filtering technique. Our result on 70 instances of NMR peak assignment shows that new method is quite promising. In particular, the method is clearly superior to the two-layer algorithm and the two approximation algorithms that we recently developed. As the framework is especially designed for problems where only limited NMR peaks are available and the peaks could be noisy, we expect that it will make a useful tool for NMR structure determination for large proteins, in conjunction with computational prediction and modeling tools. Though the development of the framework is still in its early stage, the tests have shown some highly encouraging results. In particular, using the HNCA spectrum alone for the connectivity information, or using the $^{13}C_\alpha$ chemical shifts without the $^{1}H_\alpha$ chemical shifts, we can align a significant portion of the spin systems correctly. Our planned further research can be outlined as follows.

More realistic models for treating segments. In our current model, all segments are assumed to be correct. In reality, segment determination is a nontrivial issue. Due to artificial and missing peaks, segments could be inconsistent with each other, e.g., they could overlap. In our current implementation, overlapping segments are broken into smaller pieces to avoid conflicts but some connectivity information could be lost in this way. A more general model for segments will be developed to deal with the ambiguity and conflicting information.

Improved scoring scheme. Our current scoring scheme is mainly based on resonance frequency distributions of different single amino acid types. The scoring scheme has a limited discerning power, since the correct assignment generally does not have the best score. In particular, the scheme depends on predicted secondary structures, which have about a 20% error [13]. Clearly, more work on the scoring function is needed in the future. A great deal of additional information can be used to enhance the discerning power of the scoring scheme. We will examine how environment information, say an amino acid sitting in the middle of a particular tripeptide, may help improve the scoring scheme. We are also planning to investigate how a predicted 3D structure by threading [23] can be used to help design more effective scoring schemes.

More experiments on NMR data. Our current approach involves only four different NMR spectra. A typical NMR structure determination process may use more than ten spectra. We will extend our framework to other NMR experiments, which are suitable for our goals, *i.e.* getting backbone assignments for large proteins [9]. We may include a pair of CBCANH and CBCA(CO)NH experiments to use $^{13}C_\beta$ chemical shifts for our scoring scheme. The expanded list of NMR spectral data should clearly increase the assignment specificity.

Assignments of NOEs. Another challenging issue related to the peak assignment problem is the assignment of nuclear Overhauser effects (NOEs). The quality of assigned NOEs, as well as the quantity of assigned NOEs, directly affect the quality of the protein structure. The fundamental difficulty is that it requires full knowledge of protein 3D structure. We expect that our capability of the protein structure prediction may help to solve this problem.

Acknowledgments

We thank Dongsup Kim, Victor Olman and Jane Razumovskaya of the Protein Informatics Group of Oak Ridge National Laboratory for helpful discussions and assistance. GL was supported in part by NSERC RGPIN249633 and Startup Grant REE-P5-01-02-Sci from University of Alberta. DX and YX were supported by the Office of Biological and Environmental Research, U.S. Department of Energy, under Contract DE-AC05-00OR22725, managed by UT-Battelle, LLC. ZZC was supported in part by the Grant-in-Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture of Japan, under Grant No. 12780241. TJ was supported in part by a UCR startup grant and NSF Grants CCR-9988353 and ITR-0085910. JW was supported in part by the NSF Grant CCR-9988353.

References

[1] A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. Naor, and B. Schieber. A unified approach to approximating resource

allocation and scheduling. *Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC'00)*, pages 735–744, 2000.

[2] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–46, 1985.

[3] C. Bartels, P. Güntert, M. Billeter, and K. Wüthrich. GARANT-A general algorithm for resonace assignment of multidimensional nuclear magnetic resonance spectra. *J. Comp. Chem.*, 18:139–149, 1996.

[4] C. Bartels, T. Xia, M. Billerter, P. Güntert, and K. Wüthrich. The program XEASY for computer-supported NMR spectral analysis of biological macromolecules. *J. Biomol. NMR*, 5:1–10, 1995.

[5] A. Bax and S. Grzesiek. Methodological advances in protein NMR. *Acc. Chem. Res.*, 26:131–138, 1993.

[6] R. Bernstein, C. Cieslar, A. Ross, H. Oschkinat, J. Freund, and T. A. Holak. Computer assisted assignment of multidimensional MMR-spectra of proteins: application to 3D NOESY-HMQC and TOCSY-HMQC spectra. *J. Biomol. NMR*, 3:245–251, 1993.

[7] J. Cavanagh, W. J. Fairbrother, A. G. Palmer III, and N. J. Skelton. *Protein NMR Spectroscopy: Principles and Practice*. Academic Press, San Diego, 1996.

[8] Z.-Z. Chen, T. Jiang, G.-H. Lin, J. J. Wen, D. Xu, J. Xu, and Y. Xu. Approximation algorithms for NMR spectral peak assignment. *Theoretical Computer Science*, Feb, 2002. In press.

[9] A. E. Ferentz and G. Wagner. NMR spectroscopy: a multifaceted approach to macromolecular structure. *Quart. Rev. Biophys.*, 33:29–65, 2000.

[10] A. V. Goldberg and R. Kennedy. An efficient cost scaling algorithm for the assignment problem. *Mathematical Programming*, 71, 1995.

[11] W. Gronwald, R. F. Boyko, F. D. Sönnichsen, and D. S. Wishart. ORB, a homology-based program for the prediction of protein NMR chemical shifts. *J. Biomol. NMR*, 10:165–179, 1997.

[12] P. Güntert, M. Salzmann, D. Braun, and K. Wüthrich. Sequence-specific NMR assignment of proteins by global fragment mapping with the program mapper. *J. Biomol. NMR*, 18:129–137, 2000.

[13] D. T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, 292:195–202, 1999.

[14] E. L. Lawler. *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Winston, New York, 1976.

[15] J. A. Lukin, A. P. Gove, S. N. Talukdar, and C. Ho. Automated probabilistic method for assigning backbone resonances of (^{13}C,^{15}N)-labeled proteins. *J. Biomol. NMR*, 9:151, 1997.

[16] H. N. B. Moseley and G. T. Montelione. Automated analysis of NMR assignments and structures for proteins. *Curr. Opin. Struc. Biol.*, 9:635–642, 1999.

[17] National Institute of General Medical Sciences. Pilot projects for the protein structure initiative (structural genomics). *http://www.nih.gov/grants/guide/rfa-files/RFA-GM-99-009.html*, June:RFA GM–99–009, 1999.

[18] University of Wisconsin. *BioMagResBank. http://www.bmrb.wisc.edu.* University of Wisconsin, Madison, Wisconsin, 2001.

[19] E. C. van Geerestein-Ujah, M. Slijper, R. Boelens, and R. Kaptein. Graph-theoretical assignment of secondary structure in multidimensional protein NMR spectra: Application to the *lac* repressor headpiece. *J. Biomol. NMR*, 5:67–78, 1995.

[20] M. P. Williamson and T. Asakura. Empirical comparisons of models for chemical-shift calculations in proteins. *J. Mag. Res., B*, 101:63–71, 1993.

[21] K. Wüthrich. *NMR of Proteins and Nucleic Acids*. Wiley, New York, 1986.

[22] J. Xu, S. K. Straus, B. C. Sanctuary, and L. Trimble. Use of fuzzy mathematics for complete automated assignment of peptide ^{1}H 2D NMR spectra. *J. Mag. Reson., B*, 103:53–58, 1994.

[23] Y. Xu and D. Xu. Protein threading using PROSPECT: Design and evaluation. *Proteins: Struct. Funct. Genet.*, 40:343–354, 2000.

[24] Y. Xu, D. Xu, D. Kim, V. Olman, J. Razumovskaya, and T. Jiang. Automated assignment of backbone NMR peaks using constrained bipartite matching. *IEEE Computing in Science & Engineering*, 4:50–62, 2002.

[25] D. E. Zimmerman, C. A. Kulikowski, Y. Huang, W. F. M. Tashiro, S. Shimotakahara, C. Chien, R. Powers, and G. T. Montelione. Automated analysis of protein NMR assignments using methods from artificial intelligence. *J. Mol. Biol.*, 269:592–610, 1997.