Protein Engineering vol.7 no.9 pp. 1059–1068, 1994

The protein threading problem with sequence amino acid interaction preferences is NP-complete

Richard H.Lathrop

Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

In recent protein structure prediction research there has been a great deal of interest in using amino acid interaction preferences (e.g. contact potentials or potentials of mean force) to align ('thread') a protein sequence to a known structural motif. An important open question is whether a polynomial time algorithm for finding the globally optimal threading is possible. We identify the two critical conditions governing this question: (i) variable-length gaps are admitted into the alignment, and (ii) interactions between amino acids from the sequence are admitted into the score function. We prove that if both these conditions are allowed then the protein threading decision problem (does there exist a threading with a score $\leq K$?) is NP-complete (in the strong sense, i.e. is not merely a number problem) and the related problem of finding the globally optimal protein threading is NP-hard. Therefore, no polynomial time algorithm is possible (unless P = NP). This result augments existing proofs that the direct protein folding problem is NP-complete by providing the corresponding proof for the 'inverse' protein folding problem. It provides a theoretical basis for understanding algorithms currently in use and indicates that computational strategies from other NP-complete problems may be useful for predictive algorithms.

Key words: contact potentials/inverse protein folding/NP-complete/protein structure prediction/protein threading/sequence–structure alignment

Introduction

The protein folding problem is to start from a string giving the protein's amino acid sequence and compute its correctly folded 3-D protein structure. It is an important problem because proteins underlie almost all biological processes and their function follows directly from their 3-D folded shape. Its importance is escalating rapidly due to the rapid increase in the number of sequences becoming available compared with the slow growth in the number of experimentally determined 3-D protein structures. The direct approach to protein folding seeks to find the folded conformation having minimum energy. This is difficult because (i) a folded protein results from the delicate energy balance of powerful atomic forces and (ii) the vast number of possible conformations poses a formidable computational barrier. The forces involved are often poorly understood or difficult to model accurately, and include stabilizing and destabilizing terms making large contributions of opposite sign summed over a very large number of atoms (Creighton, 1983). The computational burden of the direct approach has been shown to be NP-hard (widely assumed to require an exponential amount of time) even on simplified tion (e.g. secondary structure) constraints (Ngo and Marks, 1992; Fraenkel, 1993; Unger and Moult, 1993).

One important alternative approach is to use the known protein structures as (i) spatial folding templates, (ii) additional knowledge about protein structure and (iii) constraints on possible folds. This is a powerful strategy because folded proteins exhibit recurring patterns of organization. Chothia (1992) estimates that there are only ~1000 different protein structural families. In this approach, each known protein structure (or family) 'recognizes' the protein sequences likely to fold into a similar structure. Because it starts with structures and predicts sequences instead of starting with sequences and predicting structures, it is often referred to as the 'inverse' folding problem. In its fully general sense it includes *ab initio* design of protein sequences to achieve a target structure (Pabo, 1983), but we shall restrict attention to folding given native sequences. The known structure establishes a set of possible amino acid positions in 3-D space (perhaps the spatial locations of its main-chain α carbons). 'Recognition' is mediated by a suitable score function. An alignment between spatial positions and sequence amino acids is usually a by-product of the recognition step. The sequence is given a similar 3-D fold by placing its amino acids into their aligned spatial positions. [Further techniques are necessary to correctly place the variable loop regions (Greer, 1990; Zheng *et al.*, 1993) and position the side chains (Desmet *et al.*, 1992), but the focus of this paper is on predicting and placing the conserved fold.] The process of aligning a sequence to a structure and thereby guiding the spatial placement of sequence amino acids is referred to as 'threading' the sequence onto the structure (Bryant and Lawrence, 1993). 'A threading' means a specific alignment between sequence and structure (chosen from the large number of possible alignments). In this way 'threading' specializes the more general term 'alignment' to refer specifically to a structure (considered as a template) and a sequence (considered as being arranged on the template).

Initially, such methods employed primary sequence string similarity between the candidate sequence and the native sequence of the structure to perform the threading ('homology modeling' or 'homological extension'). Computing the sequence similarity yields a direct alignment of amino acids in the sequences of the candidate and structure (Sankof and Kruskal, 1983), and hence their spatial positions. In cases where the sequence similarity is high this is still the most successful protein structure prediction method known, but it is of limited general use because few sequences have high primary sequence similarity to another whose structure is known. Recently, however, researchers have been able to extend the match process beyond primary sequence similarity and align a sequence directly to a structure.

These new approaches exploit the fact that amino acid types have different preferences for occupying different structural environments (e.g. preferences for being in α-helices or β-

that there appear to be distinct preferences for side-chain contact (e.g. contact potentials; Maiorov and Crippen, 1992), or more generally for spatial proximity (e.g. potentials of mean force; Sippl, 1990, 1993), as a function of those environments. For example, a buried charged residue may be more likely to be adjacent to another buried residue of opposite charge. These

large space of possible solutions (the 2^N possible truth value assignments to N Boolean variables), but is easily solved in polynomial time by resolution techniques (Garey and Johnson, 1979). However, no polynomial time solution is known for any NP-complete problem.

In the remainder of this section we categorize the different

include many problems deeply central to computer science, and so a great deal of effort by a great many talented people has been expended searching for a polynomial time solution to any one of them. Because so many people have failed, it is widely accepted that no polynomial time algorithm is likely to be found.

In some cases it is possible to prove directly from first principles that a problem at hand is NP-complete, but this is usually quite difficult. Most proofs proceed by constructing a polynomial time transformation of another problem, already known to be NP-complete, into an instance of the problem at hand. It follows that if the problem at hand could be solved in polynomial time, so could the other problem, and therefore by extension all of the problems in NP. Consequently, the problem at hand is NP-complete.

NP-complete problems are phrased as decision problems to which the answer is either 'yes' or 'no.' For example, the decision problem addressed by this paper is, 'Does there exist a threading of this sequence onto this structure under this score function, such that the threading score is $\leq K$?' This might be the case in which a candidate threading has already been found and one wishes simply to ask whether or not another threading with a better score exists. For many NP-complete decision problems there is an associated optimization problem for which the task is to produce an optimal solution

We will state the protein (or motif) threading problem in sufficient generality to cover a wide range of cases. The problem and formalisms are equally applicable to threading protein core motifs or super-secondary structure motifs. Our general definition is similar to that of Bryant and Lawrence (1993). Details are contained in Lathrop and Smith (1994), from which many of the definitions here are drawn. A probability analysis appears in White *et al.* (1994). Although for generality the problem is stated in terms of core segments of contiguous amino acids, the proof actually uses segments of length exactly 1. Consequently, it applies equally well to formulations that admit gaps between any pair of amino acids (e.g. Godzik *et al.*, 1992; Jones *et al.*, 1992), provided that they model explicit sequence amino acid interactions.

Informal problem motivation

All current threading proposals replace the 3-D coordinates of the known structure by an abstract description in terms of core elements and segments, neighbor relationships, distances, environments and the like. This avoids the computational cost of full-atom molecular mechanics or dynamics (Weiner *et al.*, 1984; Brooks *et al.*, 1990) in favor of a much less detailed, and hence much faster, discrete alignment between sequence and structure. However, important aspects of protein structure (such as intercalated side-chain packing, excluded volume,

(directed) edge, and the edge is labeled with the pairwise environment. The edge in the graph corresponds to the cell in the matrix, and the edge label corresponds to the label contained in the cell. Related representations include adjacency matrix, contact graph, and so on.

In this framework, a protein core folding motif, C, consists of m core segments, C_i, each representing a set of contiguous amino acid positions. Core segments are usually the pieces of secondary structure comprising the tightly packed internal

reader interested in formal details should turn to Appendix.

The canonical (and first) NP-complete problem is SATISFIABILITY. A problem instance consists of a set of Boolean variables plus a set of Boolean clauses (a clause is a disjunction, i.e. logical OR, of a set of literals; a literal is either one of the variables or the negation of one of the variables). The question is whether any setting (truth-value assignment) of the variables makes all of the clauses true simultaneously. 3SAT is a well-known variant which restricts the clauses to contain exactly

(vii) Thus, if we could solve the general PRO-THREAD problem in polynomial time, we could solve ONE-IN-THREE 3SAT in polynomial time. PRO-THREAD is NP-complete.

In fact, PRO-THREAD is NP-complete in the strong sense (i.e. is not a number problem) because the only numbers used in the construction are 0 and 1. The optimization problem, to produce an optimal threading, is NP-hard.

Discussion

We identified the two critical conditions governing the computational complexity of protein threading as whether or not: (i) variable-length gaps are permitted in the alignments; and (ii) the score for placing a sequence amino acid into a given structure position depends on the specific amino acid types from the sequence being threaded that are placed into neighboring (interacting) structure positions.

The condition of variable-length gaps plays a different role to the condition of pairwise interactions between amino acids from the sequence being threaded. Allowing variable-length gaps ensures that the search space of possible solutions is exponentially large. This is necessary, but not sufficient, for a problem to be NP-complete. Allowing pairwise interactions

discovered, such restrictions could be reflected in specialized threading and folding algorithms.

Failing this, protein threading (and folding) algorithms can draw on a wealth of computational methods that have been developed to cope with NP-complete problems. For example, the knapsack problem (how to pack objects of different sizes and values into a finite volume so as to maximize the total contained value) is known to be NP-complete, but branch-and-bound algorithms have been so successful that many consider it to be an efficiently solvable problem, even though the algorithms involved (of course) have a formal exponential time complexity (Garey and Johnson, 1979, p. 9). Packing objects into a knapsack is analogous to packing amino acids into a protein core, and a branch-and-bound algorithm has been employed successfully to find the globally optimal threading in the general protein threading problem (Lathrop and Smith, 1994). A number of other computational techniques, such as simulated annealing and genetic algorithms, can be used to find good approximate solutions to NP-complete problems. As expected, researchers are actively pursuing each of these avenues.

The basic proof can be used to prove that many threading

approximate solution will correspond to a misfolded protein. Whether finding the globally optimal threading is necessary or not can only be answered relative to the goals that led one to attempt the threading.

Acknowledgements

The author thanks Barbara Bryant, Jim Cornette, Michael de la Maza, Ilya Muchnik, Kimmen Sjölander, Lisa Tucker-Kellog and the anonymous referees for comments that improved the manuscript, Jim White for discussions of the mathematical formalism, and Temple Smith for discussions of proteins. This report describes research performed at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology, in consortium with the BioMolecular Engineering Research Center of Boston University, sponsored by the National Science Foundation under grant DIR-9121548. Support for the Artificial Intelligence Laboratory's research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-91-J-4038. Support for the BioMolecular Engineering Research Center's research is provided in part by the National Institutes of Health under grant RR02275-05.

References

Bowie,F.U., Lüthy,R. and Eisenberg,D. (1991) *Science*, 253, 164–170.
Brooks,C.L., Karplus,M. and Pettitt,B.M. (1990) *Proteins: A Theoretical Perspective of Dynamics, Structure and Thermodynamics*. John Wiley and Sons, New York.
Bryant,S.H. and Lawrence,C.E. (1993) *Proteins: Struct. Funct. Genet.*, 16, 92–112.
Chothia,C. (1992) *Nature*, 357, 543–544.
Cook,S.A. (1971) In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*. Association for Computing Machinery, New York, pp. 151–158.
Creighton,T.E. (1983) *Biopolymers*, 22, 49.
Crippen,G.M. (1991) *Biochemistry*, 30, 4232–4237.
Desmet,J., De Maeyer,M., Hazes,B. and Lasters,I. (1992) *Nature*, 356, 539–542.
Finkelstein,A.V. and Reva,B. (1991) *Nature*, 351, 497–499.
Fraenkel,A.S. (1993) *Bull. Math. Biol.*, 55, 1199–1210.
Garey,M.R. and Johnson,D.S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H.Freeman and Company, New York.
Godzik,A., Kolinski,A. and Skolnick,J. (1992) *J. Mol. Biol.*, 227, 227–238.
Goldstein,R.A., Luthey-Schulten,Z.A. and Wolynes,P.G. (1992) *Proc. Natl Acad. Sci. USA*, 89, 9029–9033.
Greer,J. (1990) *Proteins: Struct. Funct. Genet.*, 7, 317–333.
Hendlich,M., Lackner,P., Weitckus,S., Floeckner,H., Froschauer,R., Gottsbacher,K., Casari,G. and Sippl,M.J. (1990) *J. Mol. Biol.*, 216, 167–180.
Hopcroft,J.E. and Ullman,J.D. (1979) *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, MA.
Johnson,M.S., Overington,J.P. and Blundell,T.L. (1993) *J. Mol. Biol.*, 231, 735–752.
Jones,D.T., Taylor,W.R. and Thornton,J.M. (1992) *Nature*, 358, 86–89.
Lathrop,R.H. and Smith,T.F. (1994) *Proceedings of the 27th Hawaii International Conference on System Sciences*. IEEE Computer Society Press, Los Alamitos, CA, pp. 365–374.
Lewis,H.R. and Papadimitriou,C.H. (1981) *Elements of the Theory of Computation*. Prentice-Hall, Englewood Cliffs, NJ.
Lüthy,R., Bowie,J.U. and Eisenberg,D. (1992) *Nature*, 356, 83–85.
Maiorov,V.N. and Crippen,G.M. (1992) *J. Mol. Biol.*, 227, 876–888.
Miyazawa,S. and Jernigan,R.L. (1985) *Macromolecules*, 18, 534–552.
Ngo,J.T. and Marks,J. (1992) *Protein Engng*, 5, 313–321.
Orengo,C.A. and Taylor,W.R. (1990) *J. Theor. Biol.*, 147, 517–551.
Ouzounis,C., Sander,C., Scharf,M. and Schneider,R. (1993) *J. Mol. Biol.*, 232, 805–825.
Pabo,C. (1983) *Nature*, 301, 200.
Sankoff,D. and Kruskal,J.B. (1983) *Time Warps, String Edits and Macromolecules*. Addison-Wesley, Reading, MA.
Schaefer,T.J. (1978) *Proceedings of the 10th Annual ACM Symposium on Theory of Computing*. Association for Computing Machinery, New York, pp. 216–226.
Sippl,M.J. (1990) *J. Mol. Biol.*, 213, 859–883.
Sippl,M.J. (1993) *J. Comput.-Aided Mol. Design*, 7, 473–501.
Sippl,M.J. and Weitckus,S. (1992) *Proteins: Struct. Funct. Genet.*, 13, 258–271.
Taylor,W.R. and Orengo,C.A. (1989) *J. Mol. Biol.*, 208, 1–22.
Unger,R. and Moult,J. (1993) *Bull. Math. Biol.*, 55, 1183–1198.
Weiner,S.J., Kollman,P.A., Case,D.A., Singh,U.C., Ghio,C., Alagona,G., Profeta,S. and Weiner,P. (1984) *J. Am. Chem. Soc.*, 106, 765–784.
White,J., Muchnik,I. and Smith,T.F. (1994) *Math. Biosci.*, in press.
Wilmanns,M. and Eisenberg,D. (1993) *Proc. Natl Acad. Sci. USA*, 90, 1379–1383.
Zheng,Q., Rosenfeld,R., Vajda,S. and DeLisi,C. (1993) *Protein Sci.*, 2, 1242–1248.

Received January 3, 1994; revised March 28, 1994; accepted May 16, 1994

Appendix

A formal problem statement and proof

Here we give a formal statement of the PRO-THREAD problem and state a formal proof that it is NP-complete.

A formal problem statement (PRO-THREAD)

Table I summarizes the notational usage of this paper.

Let G be a labeled directed graph having a set of vertices $V = (v_1, v_2, \ldots, v_p)$, a set of edges $E = (e_1, e_2, \ldots, e_q)$, a set of vertex labels L_v and a set of edge labels L_e. Let s map vertices to vertex labels and edges to edge labels. Let $C = (C_1, C_2, \ldots, C_m)$ be a partition of V. Let $C_{i,j}$ denote the jth

Table I. Notational usage of this paper

Notation	Usage
$\mathbf{a}$	a sequence of characters drawn from AA; the concatenation of $\mathbf{a}_B$ and $\mathbf{a}_U$
$\bar{\mathbf{a}}$	a subsequence of $\mathbf{a}$
AA	an alphabet of 20 characters (amino acids)
$\mathbf{a}_B$	g instances of the sequence (Q, R, S), concatenated together
$\mathbf{a}_U$	h instances of the sequence (T, F), concatenated together
B	a set of Boolean clauses (disjunctions); b_k is the kth clause
C	a set of core segments; a partition of V
C_i	a core segment; the ith element of C; a subset of V
$C_{i,j}$	a core element; the jth element of C_i; another indexing of a vertex of V
c_i	$\lvert C_i \rvert$, the length of the ith core segment
E	the set of edges of the graph G; e_i is the ith edge
E_k	a subset of E; those edges whose tail is v_k
F	the amino acid phenylalanine
f	a function mapping an m-tuple of integers to a number
f_e	a function mapping an edge and an m-tuple of integers to a number
f_l	a function mapping an integer and an m-tuple of integers to a number
f_v	a function mapping a vertex and an m-tuple of integers to a number
G	a labeled, directed graph
g	$\lvert B \rvert$, the number of Boolean clauses
h	$\lvert U \rvert$, the number of Boolean variables
K	a fixed number
L_e	the set of edge (environment) labels of the graph G
L_v	the set of vertex (environment) labels of the graph G
m	$\lvert C \rvert$, the number of core segments
n	$\lvert \mathbf{a} \rvert$, the length of the sequence $\mathbf{a}$
Q, R, S, T	the amino acids glutamine, arginine, serine, threonine
s	a function mapping vertices to vertex labels and edges to edge labels
t	a threading; an m-tuple of integers; t_i is the ith coordinate
U	a set of Boolean variables; u_i is the ith variable
V	the set of vertices of the graph G; v_i is the ith vertex
z	a literal from a clause of B
α	α_1 = Q, α_2 = R, α_3 = S
π	a function mapping a vertex and an m-tuple of integers to a character
σ_e	a function mapping two characters and an edge label to a number
σ_l	a function mapping a subsequence of $\mathbf{a}$ to a number
σ_v	a function mapping a character and a vertex label to a number

element of C_i (yielding a second indexing of V), and let $c_i = |C_i|$. For $e \in E$ let *head* (e) [respectively *tail* (e)] return the vertex at the head (respectively tail) of e.

Let AA be an alphabet of 20 characters (amino acids) and let $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ be a sequence of n characters drawn from AA.

Let $\mathbf{t} = (t_1, t_2, \ldots, t_m)$ be an m-tuple such that $1 \leq t_1$, that $t_i + c_i \leq t_{i+1}$ for $1 \leq i < m$, and that $t_m + c_m \leq n + 1$. The m-tuple $\mathbf{t}$ specifies a threading.

Let $\pi(v, \mathbf{t})$ be a function that maps vertices to characters in $\mathbf{a}$ according to $\mathbf{t}$, defined for $v = C_{i,j}$ as $\pi(v, \mathbf{t}) = a_{t_i + j - 1}$. The function π yields the sequence character (amino acid) threaded into vertex v by $\mathbf{t}$.

Let $\sigma_v(a, d)$ map the character $a \in$ AA and the vertex label $d \in L_v$ to a number. Let $\sigma_e(a, b, d)$ map the characters $a, b \in$ AA and the edge label $d \in L_e$ to a number. If $\tilde{\mathbf{a}}$ is any contiguous subsequence of $\mathbf{a}$, let $\sigma_l(\tilde{\mathbf{a}})$ map $\tilde{\mathbf{a}}$ to a number. These are score functions for vertices (amino acids), edges (interacting amino acid pairs) and loop regions (gaps), respectively.

Extend these functions to a score function f mapping $\mathbf{t}$ to a number as follows. Let:

$$f_v(v, \mathbf{t}) = \sigma_v(\pi(v, \mathbf{t}), s(v)), \tag{1}$$

$$f_e(e, \mathbf{t}) = \sigma_e(\pi(head\ (e), \mathbf{t}), \pi(tail\ (e), \mathbf{t}), s(e)), \tag{2}$$

$$f_l(i, \mathbf{t}) = \sigma_l((a_1, a_2, \ldots, a_{t_1 - 1})), \text{ if } i = 0,$$

SATISFIABILITY
↓
3SAT
↓
ONE-IN-THREE 3SAT
↓
Protein Threading Decision Problem (PRO-THREAD)
↓
Optimal Protein Threading Problem

Fig. 1. The sequence of problem transformations we employ.

$$U = \{u_1, u_2, u_3, u_4\}$$
$$B = \{(u_1, \bar{u}_2, u_3), (\bar{u}_2, u_3, \bar{u}_4), (u_1, \bar{u}_3, u_4)\}$$

Fig. 2. Example of the ONE-IN-THREE 3SAT problem.

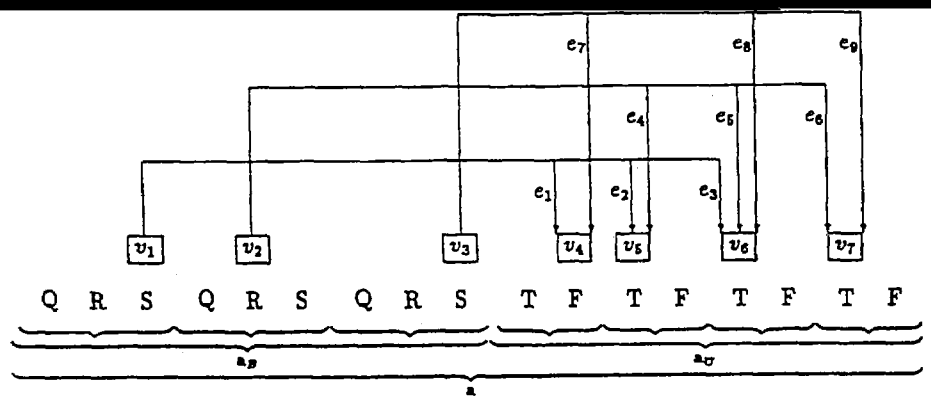
a non-deterministic algorithm need only guess a particular threading and check in polynomial time whether its score is K or less.

We transform ONE-IN-THREE 3SAT to PRO-THREAD. Figures 2–5 illustrate the transformation for a simple example.

Let $B = \{b_1, b_2, \ldots, b_g\}$ be a collection of clauses with $|b_i| = 3$, $1 \leq i \leq g$, and let $U = \{u_1, u_2, \ldots, u_h\}$ be the set of Boolean variables mentioned in B. These make up an

$$U = \{u_1 = False, u_2 = True, u_3 = True, u_4 = True\}$$
$$B = \{(False, False, True), (False, True, False), (False, False, True)\}$$

Fig. 3. Truth value assignments solving the example ONE-IN-THREE 3SAT problem.



e	*tail* (e)	*head* (e)	s(e)
e_1	v_1	v_4	$(1, P)$
e_2	v_1	v_5	$(2, N)$
e_3	v_1	v_6	$(3, P)$
e_4	v_2	v_5	$(1, N)$
e_5	v_2	v_6	$(2, P)$
e_6	v_2	v_7	$(3, N)$
e_7	v_3	v_4	$(1, P)$
e_8	v_3	v_6	$(2, N)$
e_9	v_3	v_7	$(3, P)$

The equivalence between ONE-IN-THREE 3SAT clauses and variables and their corresponding PRO-THREAD vertices is

$$\{(b_1, v_1), (b_2, v_2), (b_3, v_3), (u_1, v_4), (u_2, v_5), (u_3, v_6), (u_4, v_7), \}.$$

Fig. 4. PRO-THREAD problem equivalent to the example ONE-IN-THREE 3SAT problem.

$$t = (3, 5, 9, 11, 12, 14, 16)$$

The threading t corresponds to the placement of v_i in Figure 4. The truth-values in Figure 3, which solve the ONE-IN-THREE 3SAT problem in Figure 2, can be read directly from this threading. Note that other threadings (e.g., $(3, 8, 9, 11, 12, 14, 16)$) would yield the same truth-values.

Fig. 5. Threading solving the equivalent PRO-THREAD problem.

(j, P) if positive, $1 \leq j \leq 3$ and $1 \leq k \leq g$. Let all vertex label. If $d = (j, P)$, corresponding to the jth literal being

problem has a solution then the exhibited threading problem has a threading with a score of 0. Let the variables in U be assigned the solution truth values. We will show that $f(\mathbf{t}) = 0$ where $\mathbf{t} = (t_1, t_2, \ldots, t_{g+h})$ and:

$t_i =$

$3(i-1) + l$, if $1 \leq i \leq g$ [where $l \in \{1, 2, 3\}$ is the index of the unique literal which was satisfied in b_i], (8)

$3g + 2(i-g) - 1$, if $g < i \leq g + h$ and u_{i-g} = TRUE,

$3g + 2(i-g)$, if $g < i \leq g + h$ and u_{i-g} = FALSE.

Consider an arbitrary vertex, v_k, such that $1 \leq k \leq g$. By construction, this corresponds to b_k. Let $l \in \{1, 2, 3\}$ index the unique literal satisfying b_k under the truth values assigned to U. Since $t_k = 3(k-1) + l$ by assumption, $\pi(v_k, \mathbf{t}) = \alpha_l$ by the structure of a.

z negated?	$j = l$?	$s(e)$	$\pi(v_k, \mathbf{t}) = \alpha_l$?	$\pi(v_{g-i}, \mathbf{t})$	t.v. of u_i	t.v. of z
no	yes	(j, P)	yes	T	TRUE	TRUE
no	no	(j, P)	no	F	FALSE	FALSE
yes	no	(j, N)	no	T	TRUE	FALSE
yes	yes	(j, N)	yes	F	FALSE	TRUE

It is easy to see that z is true exactly when $l = j$. By construction this is true for exactly one edge in E_k, and consequently b_k has exactly one true literal under this truth value assignment.

($\Leftrightarrow$) It is easy to verify that the transformation can be accomplished in polynomial time. Consequently, the original