Exploiting Model Checking in Constraint-based Approaches to the Protein Folding Problem

Elisabetta De Maria Agostino Dovier
Angelo Montanari Carla Piazza

Department of Mathematics and Computer Science
University of Udine, Italy

WCB 2006
September 25th, 2006
Nantes, France

Outline of the presentation

- The Protein Folding Problem
- Protein Models
- Model Checking and Temporal Logics
- Model Checking for Protein Folding
- Some preliminary experimental results
- Future developments

The Protein Folding Problem (PFP)

The Protein Folding Problem

Given the *Primary Structure* of a protein, the PSP/PF Problem consists in determining its *Tertiary Structure* (*conformation*).

Anfinsen's thermodynamic hypothesis

The conformation adopted by a protein (*native conformation*) is the one with minimum energy.

Energy functions

The energy of a conformation can be modeled by means of *energy functions*, which express the energy level in terms of the interactions between pairs of amino acids.

Simplified models of proteins

Restrict the admissible positions of amino acids in the space

- lattice space models.

Simplified energy functions

- HP model
- 20×20 potential matrix
- HPNX model

Folding in a square lattice

Definition [Folding]

In $\mathbb{Z}^2$, a *folding* of a sequence $s = s_0 \dots s_n$ is a function $\omega : [0 \dots n] \to \mathbb{Z}^2$ such that

(i) $\forall 0 \leq i < n$, $|\omega(i) - \omega(i+1)| = 1$;

(ii) $\forall i \neq j, \omega(i) \neq \omega(j)$ (ω is self avoiding).

Definition [Topological neighbors]

Two amino acids s_i and s_j of a given folding ω are *topological neighbors* if $j \neq i \pm 1$ and $|\omega(i) - \omega(j)| = 1$.

Energy of a folding

HP: opposite of the number of topological HH neighbors.
20 $\times$ 20: summation of the contribution of each pair of topological neighbors.

Basic assumptions

- The length of a sequence $s = s_0 \dots s_n$ is n.
- To represent the conformations of a sequence of length n, we use the subset $\mathcal{L} = \{(i,j) : i \in [0,2n], j \in [0,2n]\}$ of $\mathbb{N}^2$.
- W.l.o.g., we assume $\omega(0) = (n,n)$ and we fix $\omega(1) = (n,n+1)$.
- We represent a folding of a sequence of length n as a string of length $n-1$ on the alphabet $\{l,f,r\}$.

An example

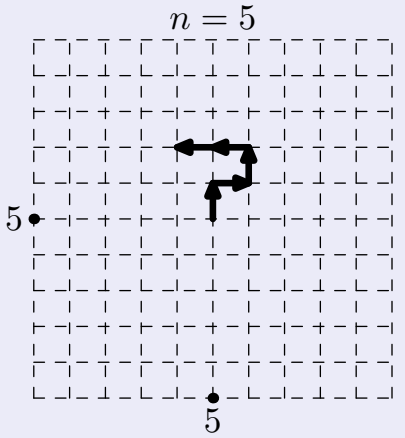


Figure: String *rllf* on 10×10 lattice.

The number of all possible foldings of a sequence of length n is bounded by 3^{n-1} (it is $\sim \frac{1.93 \cdot 2.64^n \cdot n^{0.34}}{4}$).

Valid transformations among foldings

Definition [Pivot Move]

Let $f = f_2 \ldots f_n$, with $f_i \in \{l, f, r\}$ for all $2 \leq i \leq n$, be a folding of a sequence s of length n. A folding f' of s is obtained from f through a *pivot move* with pivot $k-1$, with $2 \leq k \leq n$, if $f'_i = f_i$ for all $i \neq k$ and $f'_k \neq f_k$.

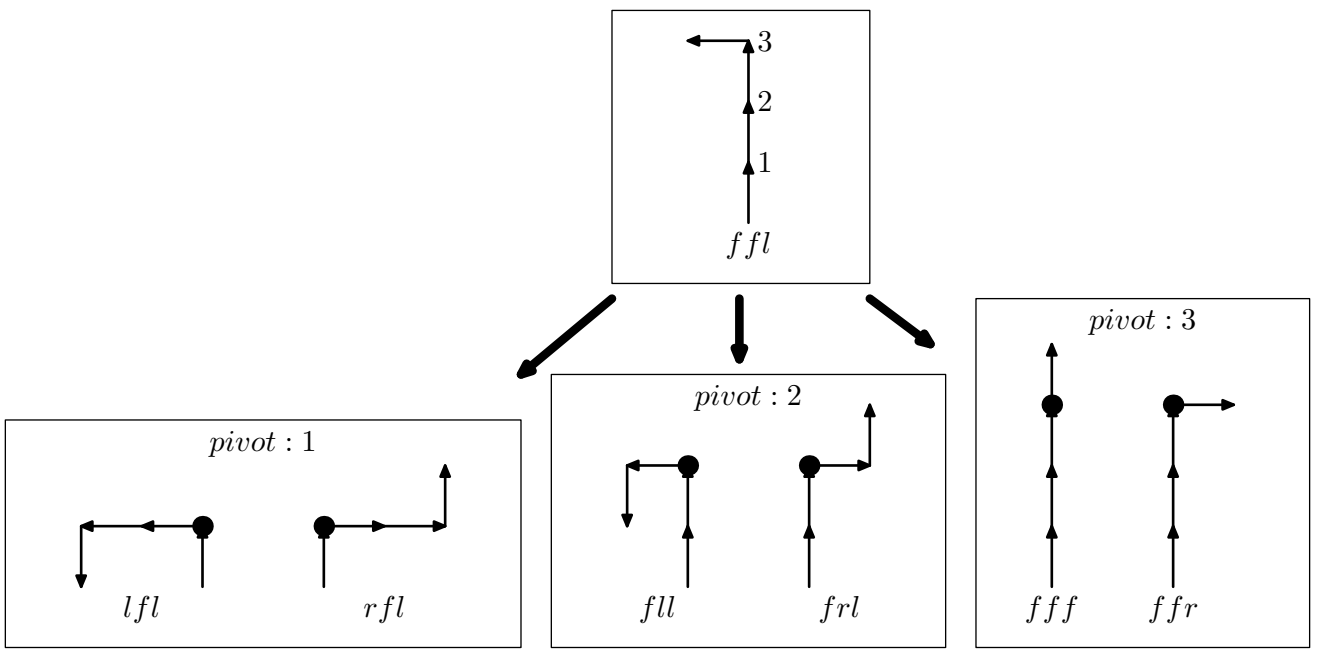


Figure: Pivot moves from string *ffl*.

Transition Systems

Definition [Transition System]

Let AP be a set of atomic propositions. A *transition system* over AP is a tuple $M = (Q, T, L)$, where

- Q is a finite set of states;
- $T \subseteq Q \times Q$ is a total transition relation, that is, for every state $q \in Q$ there is a state $q' \in Q$ such that $T(q, q')$;
- $L : Q \to 2^{AP}$ is a labeling function that maps every state into the set of atomic propositions that hold at it.

2D Protein Transition System

The 2D Protein Transition System of a string P of length n over $\{H, P\}$ is a tuple $M_P = (Q, T, L)$, where

- Q is the set of all foldings of length n on the $2n \times 2n$ 2D lattice;
- $T \subseteq Q \times Q$ contains the pairs of states (q_1, q_2) such that q_2 can be obtained from q_1 by a pivot move;
- $L : Q \to 2^{AP}$ is a labeling function over the set AP of atomic propositions which consists of the following 3(n-1) predicates

$$2nd\_l, \ldots, ith\_l, \ldots, nth\_l,$$
$$2nd\_f, \ldots, ith\_f, \ldots, nth\_f,$$
$$2nd\_r, \ldots, ith\_r, \ldots, nth\_r,$$

plus the following three predicates

$$min\_en, \quad inter\_en, \quad max\_en.$$

Temporal Logics

- Temporal logics are formalisms for describing sequences of transitions between states.
- CTL* (*computation tree logic*).
- CTL* formulae are obtained by (repeatedly) applying Boolean connectives ($\wedge, \vee, \neg, \rightarrow$), *path quantifiers* ($A, E$), and *state quantifiers* (X, U, F, G) to atomic formulae.

Path quantifiers

A: all paths starting from a given state have some property.

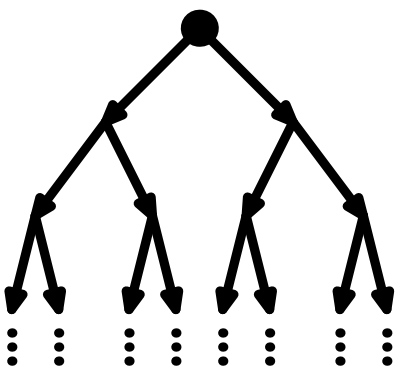


Figure: *Af*

Temporal Logics

- Temporal logics are formalisms for describing sequences of transitions between states.
- CTL* (*computation tree logic*).
- CTL* formulae are obtained by (repeatedly) applying Boolean connectives ($\wedge, \vee, \neg, \rightarrow$), *path quantifiers* ($A, E$), and *state quantifiers* (X, U, F, G) to atomic formulae.

Path quantifiers

E: some path starting from a given state has some property.

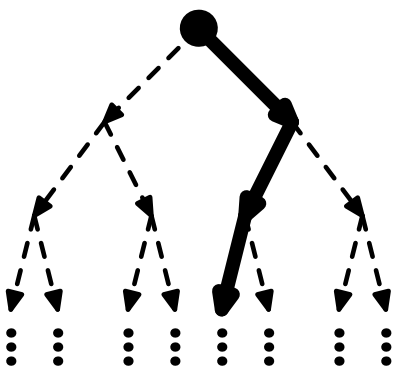


Figure: *Ef*

Temporal Logics

- Temporal logics are formalisms for describing sequences of transitions between states.
- CTL* (*computation tree logic*).
- CTL* formulae are obtained by (repeatedly) applying Boolean connectives ($\wedge, \vee, \neg, \rightarrow$), *path quantifiers* ($A, E$), and *state quantifiers* (X, U, F, G) to atomic formulae.

State quantifiers

X (*next time*): a property holds at the next state of a path.

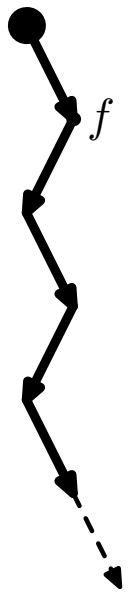


Figure: Xf

Temporal Logics

- Temporal logics are formalisms for describing sequences of transitions between states.
- CTL* (*computation tree logic*).
- CTL* formulae are obtained by (repeatedly) applying Boolean connectives ($\wedge, \vee, \neg, \rightarrow$), *path quantifiers* ($A, E$), and *state quantifiers* (X, U, F, G) to atomic formulae.

State quantifiers

U (*until*): there is a state on the path where the second of its argument properties holds and, at every preceding state on the path, the first of its two argument properties holds.

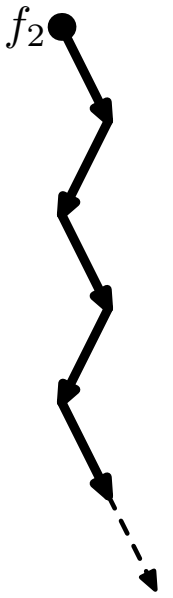


Figure: f_1 U f_2

Temporal Logics

- Temporal logics are formalisms for describing sequences of transitions between states.
- CTL* (*computation tree logic*).
- CTL* formulae are obtained by (repeatedly) applying Boolean connectives ($\wedge, \vee, \neg, \rightarrow$), *path quantifiers* ($A, E$), and *state quantifiers* (X, U, F, G) to atomic formulae.

State quantifiers

U (*until*): there is a state on the path where the second of its argument properties holds and, at every preceding state on the path, the first of its two argument properties holds.

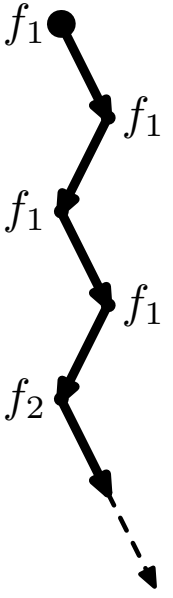


Figure: f_1 U f_2

Temporal Logics

- Temporal logics are formalisms for describing sequences of transitions between states.
- CTL* (*computation tree logic*).
- CTL* formulae are obtained by (repeatedly) applying Boolean connectives ($\wedge, \vee, \neg, \rightarrow$), *path quantifiers* ($A, E$), and *state quantifiers* (X, U, F, G) to atomic formulae.

State quantifiers

F (*sometimes in the future*): a property holds at some state on the path.

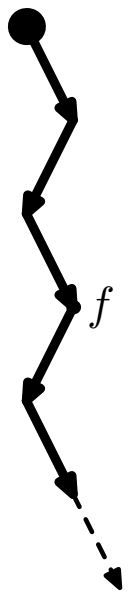


Figure: Ff

Temporal Logics

- Temporal logics are formalisms for describing sequences of transitions between states.
- CTL* (*computation tree logic*).
- CTL* formulae are obtained by (repeatedly) applying Boolean connectives ($\wedge, \vee, \neg, \rightarrow$), *path quantifiers* ($A, E$), and *state quantifiers* (X, U, F, G) to atomic formulae.

State quantifiers

G (*always in the future*): a property is true at every state on the path.

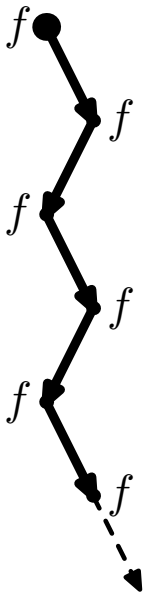


Figure: *Gf*

CTL and LTL (1)

We focus our attention on two proper fragments of CTL*.

CTL (branching time logic)

- It allows one to quantify over the paths starting from a given state.
- It constrains every state quantifier to be immediately preceded by a path quantifier.

LTL (linear time logic)

- It allows to describe events along a single computation path.
- Its formulae are of the form $\mathbf{A}f$, where f does not contain path quantifiers, but it allows the nesting of state quantifiers.

CTL and LTL (2)

CTL (branching time logic)

- The complexity of model checking is linear in the number of states and edges of the transition system.
- There are many tools for checking if finite state systems satisfy CTL formulae.

LTL (linear time logic)

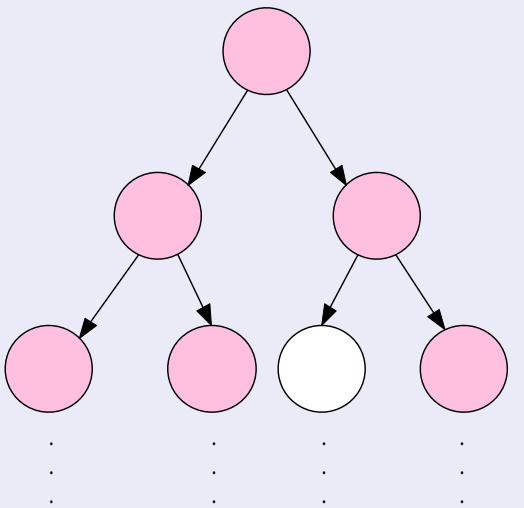
- The model checking problem is PSPACE-complete.
- To master the complexity of LTL model checking: on-the-fly model checking.

The Model Checking Problem

- Given a Transition System $M = (Q, T, L)$, a state $q \in Q$, and a temporal logic formula f expressing some desirable property of the system, the *model checking problem* consists in establishing whether $M, q \models f$ or not.

If q does not satisfy the formula, model checking algorithms produce a counterexample that falsifies it.

AG pink?



Properties of the 2D Protein Transition System (1)

F1: Does it exist a path of length at most k that reaches a state with minimum energy?

CTL: $min\_en \vee EXmin\_en \vee \cdots \vee \underbrace{EX \ldots EX}_{k} min\_en \equiv$

$\bigvee_{i=0}^{k} E_1 X_1 \ldots E_i X_i min\_en.$

LTL:

$A(\neg min\_en \wedge X \neg min\_en \wedge XX \neg min\_en \wedge \cdots \wedge \underbrace{X \ldots X}_{k} \neg min\_en)$

$\equiv A(\bigwedge_{i=0}^{k} X_1 \ldots X_i \neg min\_en).$

Properties of the 2D Protein Transition System (2)

F2: Is energy the minimum one? Alternatively, if energy is the maximum one, is it possible to reach a state with minimum energy without passing through states with intermediate energy?

CTL, LTL: $A(max\_en\ U\ min\_en)$.

Properties of the 2D Protein Transition System (3)

F3: Is it possible to reach in one step a folding where the first half of the sequence is a helix of the form *rrllrr* . . . ?

If $m = \lfloor n/2 \rfloor$ is odd, we have:
CTL: $EX(\bigwedge_{i=2,i=2+4\cdot j,j\geq 0}^{m-1}(ith\_r \wedge i+1th\_r) \wedge \bigwedge_{i=4,i=4+4\cdot j,j\geq 0}^{m-1}(ith\_l \wedge i+1th\_l))$.
If $m = 2+4\cdot j, j \geq 0$, we have:
CTL: $EX(\bigwedge_{i=2,i=2+4\cdot j,j\geq 0}^{m-1}(ith\_r \wedge i+1th\_r) \wedge \bigwedge_{i=4,i=4+4\cdot j,j\geq 0}^{m-1}(ith\_l \wedge i+1th\_l) \wedge mth\_r)$.
If $m = 4+4\cdot j, j \geq 0$, we have:
CTL: $EX(\bigwedge_{i=2,i=2+4\cdot j,j\geq 0}^{m-1}(ith\_r \wedge i+1th\_r) \wedge \bigwedge_{i=4,i=4+4\cdot j,j\geq 0}^{m-1}(ith\_l \wedge i+1th\_l) \wedge mth\_l)$.

Properties of the 2D Protein Transition System (4)

F4: Is it true that every state which is at most k steps far from the current one has maximum energy, i.e., energy equal to 0?

CTL: $max\_en \wedge AXmax\_en \wedge \cdots \wedge \underbrace{AX \ldots AX}_{k} max\_en \equiv$

$\bigwedge_{i=0}^{k} A_1 X_1 \ldots A_i X_i max\_en.$

LTL: $A(max\_en \wedge Xmax\_en \wedge \cdots \wedge \underbrace{X \ldots X}_{k} max\_en) \equiv$

$A(\bigwedge_{i=0}^{k} X_1 \ldots X_i max\_en).$

Experimental results (1)

SICStus Prolog

- 2D Protein Transition System
- Model Checking algorithms to verify properties F1-F4 and other relevant properties.

Example

States with energy equal to 0 that satisfy property $F1$ when $k = 1$, i.e., states with maximum energy that reach in one step a state with minimum energy.

Experimental results (2)

N=8

HP model

string= *HHHHHHHHH*;
min_en=-4;
States fullfiling the request (8):
$lrflflf \rightarrow llflflf$, $lfflflf \rightarrow llflflf$,
$rlfrfrf \rightarrow rrfrfrf$, $rffrfrf \rightarrow rrfrfrf$,
$flflfrl \rightarrow flflfll$, $flflffl \rightarrow flflfll$,
$frfrflr \rightarrow frfrfrr$, $frfrffr \rightarrow frfrfrr$.

20×20 model

string= *cilfmvwhy*;
min_en=-8179;
States fullfiling the request (4):
$lrflflf \rightarrow llflflf$, $lfflflf \rightarrow llflflf$,
$rlfrfrf \rightarrow rrfrfrf$, $rffrfrf \rightarrow rrfrfrf$.

Experimental results (3)

Example

Are there states with an energy different from the minimum one that may reach in one step a state with a greater energy which, in its turn, may reach in a few steps a state with minimum energy?

HP model

N=7;
string= *HHHHHHHH*;
min_en=-3;
$lrlfll(-2) \rightarrow lrlffl(0) \rightarrow lrllfl(-3)$.

20×20 model

N= 5;
string= *pcdehw*;
min_en=-2777;
$flrr(-613) \rightarrow flrf(0) \rightarrow fllf(-2777)$.

Ongoing work and future developments

State explosion problem: a protein of length n gives rise to a transition system where the number of states is $\Theta(3^{n-1})$.

$\Rightarrow$

On the fly Model Checking.

Improve the solution search of the protein folding problem.

Understand protein energy functions.

Ongoing work and future developments

State explosion problem: a protein of length n gives rise to a transition system where the number of states is $\Theta(3^{n-1})$.

$\Rightarrow$

On the fly Model Checking.

Improve the solution search of the protein folding problem.

Understand protein energy functions.

Ongoing work and future developments

State explosion problem: a protein of length n gives rise to a transition system where the number of states is $\Theta(3^{n-1})$.

$\Rightarrow$

On the fly Model Checking.

Improve the solution search of the protein folding problem.

Understand protein energy functions.


THANKS


UNIVERSITAS STUDIORUM UTINENSIS