Application: Protein Structure Prediction
The problem

**IN:** sequence $s$ in $\{H, P\}^n$

$$\text{HHPPPHHPHHPPPHHHPPPHHPPPHPPPHH}$$

**OUT:** self avoiding walk $\omega$ on cubic/fcc lattice with minimal HP-energy $E_{HP}(s, \omega)$
A First Constraint Model

- Variables $X_1, \ldots, X_n$, $Y_1, \ldots, Y_n$, $Z_1, \ldots, Z_n$ and HHContacts

$$
\begin{pmatrix}
X_i \\
Y_i \\
Z_i
\end{pmatrix}
is the position of the $i$th monomer $\omega(i)$

- Domains

$$D(X_i) = D(Y_i) = D(Z_i) = \{-n, \ldots, n\}$$

- Constraints
  1. positions $i$ and $i + 1$ are neighbored (chain)
  2. all positions differ (self-avoidance)
  3. relate HHContacts to $X_i, Y_i, Z_i$
  4. $\begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
A First Constraint Model

- Variables $X_1, \ldots, X_n, Y_1, \ldots, Y_n, Z_1, \ldots, Z_n$ and $HHContacts$

$$\begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix}$$ is the position of the $i$th monomer $\omega(i)$

- Domains

$$D(X_i) = D(Y_i) = D(Z_i) = \{-n, \ldots, n\}$$

- Constraints
  1. positions $i$ and $i + 1$ are neighbored (chain)
  2. all positions differ (self-avoidance)
  3. relate $HHContacts$ to $X_i, Y_i, Z_i$

$$\begin{pmatrix} X_1 \\ Y_1 \\ Z_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$
A First Constraint Model

- **Variables** $X_1, \ldots, X_n, Y_1, \ldots, Y_n, Z_1, \ldots, Z_n$ and $HHContacts$

  $$
  \begin{pmatrix}
  X_i \\
  Y_i \\
  Z_i
  \end{pmatrix}
  $$

  is the position of the $i$th monomer $\omega(i)$

- **Domains**

  $$
  D(X_i) = D(Y_i) = D(Z_i) = \{-n, \ldots, n\}
  $$

- **Constraints**

  1. positions $i$ and $i + 1$ are neighbored (chain)

  2. all positions differ (self-avoidance)

  3. relate $HHContacts$ to $X_i, Y_i, Z_i$

  $$
  \begin{pmatrix}
  X_1 \\
  Y_1 \\
  Z_1
  \end{pmatrix} =
  \begin{pmatrix}
  0 \\
  0 \\
  0
  \end{pmatrix}
  $$

  4. $$
  \begin{pmatrix}
  X_1 \\
  Y_1 \\
  Z_1
  \end{pmatrix} =
  \begin{pmatrix}
  0 \\
  0 \\
  0
  \end{pmatrix}
  $$
The First Model in More Detail (Cubic Lattice)

The Constraints cannot be expressed directly, i.e. we need auxiliary variables

\[ X_{\text{diff}}_{ij} = |X_i - X_j| \quad Y_{\text{diff}}_{ij} = |Y_i - Y_j| \quad Z_{\text{diff}}_{ij} = |Z_i - Z_j| \]

1. Positions \( i \) and \( i + 1 \) neighbored (chain)

\[ X_{\text{diff}}_{i(i+1)} + Y_{\text{diff}}_{i(i+1)} + Z_{\text{diff}}_{i(i+1)} = 1 \]

2. All positions differ (self-avoidance)

\[ X_{\text{diff}}_{ij} + Y_{\text{diff}}_{ij} + Z_{\text{diff}}_{ij} \neq 0 \quad (\text{for } i \neq j). \]

3. Relate \( HH\text{Contacts} \) to \( X_i, Y_i, Z_i \)
   Detect HH-contact, if \( X_{\text{diff}}_{ij} + Y_{\text{diff}}_{ij} + Z_{\text{diff}}_{ij} = 1 \) for \( s_i = s_j = H \). Then add 1 to \( HH\text{Contacts} \).
   (Technically, use reified constraints)
The First Model in More Detail (Cubic Lattice)

The Constraints cannot be expressed directly, i.e. we need auxiliary variables

\[
X_{\text{diff}ij} = |X_i - X_j| \quad Y_{\text{diff}ij} = |Y_i - Y_j| \quad Z_{\text{diff}ij} = |Z_i - Z_j|
\]

1. Positions \( i \) and \( i + 1 \) neighbored (chain)

\[
X_{\text{diff}i(i+1)} + Y_{\text{diff}i(i+1)} + Z_{\text{diff}i(i+1)} = 1
\]

2. All positions differ (self-avoidance)

\[
X_{\text{diff}ij} + Y_{\text{diff}ij} + Z_{\text{diff}ij} \neq 0 \quad (\text{for } i \neq j).
\]

3. Relate \( \text{HHContacts} \) to \( X_i, Y_i, Z_i \)

Detect HH-contact, if \( X_{\text{diff}ij} + Y_{\text{diff}ij} + Z_{\text{diff}ij} = 1 \) for \( s_i = s_j = H \). Then add 1 to \( \text{HHContacts} \).

(Technically, use reified constraints)
The First Model in More Detail (Cubic Lattice)

The Constraints cannot be expressed directly, i.e. we need auxiliary variables

\[ X_{diff_{ij}} = |X_i - X_j| \quad Y_{diff_{ij}} = |Y_i - Y_j| \quad Z_{diff_{ij}} = |Z_i - Z_j| \]

1. Positions \( i \) and \( i + 1 \) neighbored (chain)

\[ X_{diff_{i(i+1)}} + Y_{diff_{i(i+1)}} + Z_{diff_{i(i+1)}} = 1 \]

2. All positions differ (self-avoidance)

\[ X_{diff_{ij}} + Y_{diff_{ij}} + Z_{diff_{ij}} \neq 0 \quad \text{(for } i \neq j \text{).} \]

3. Relate \( HHContacts \) to \( X_i, Y_i, Z_i \)

Detect HH-contact, if \( X_{diff_{ij}} + Y_{diff_{ij}} + Z_{diff_{ij}} = 1 \) for \( s_i = s_j = H \). Then add 1 to \( HHContacts \).

(technically, use \textit{reified constraints})
• Model is a COP (Constraint Optimization Problem)
• Branch and Bound Search for Minimizing Energy
• Combined with Symmetry Breaking
• How good is the propagation?
• Main problem of propagation: bounds on contacts/energy
  From a partial solution, the solver cannot estimate the
  maximally possible number of HH-contacts well.
Solving the First Model

- Model is a COP (Constraint Optimization Problem)
- Branch and Bound Search for Minimizing Energy
- Combined with Symmetry Breaking
- How good is the propagation?
- Main problem of propagation: bounds on contacts/energy
  From a partial solution, the solver cannot estimate the maximally possible number of HH-contacts well.
Solving the First Model

- Model is a COP (Constraint Optimization Problem)
- Branch and Bound Search for Minimizing Energy
- Combined with Symmetry Breaking

- How good is the propagation?
- Main problem of propagation: bounds on contacts/energy
  From a partial solution, the solver cannot estimate the maximally possible number of HH-contacts well.
Solving the First Model

- Model is a COP (Constraint Optimization Problem)
- Branch and Bound Search for Minimizing Energy
- Combined with Symmetry Breaking
- How good is the propagation?
  - Main problem of propagation: bounds on contacts/energy
    From a partial solution, the solver cannot estimate the maximally possible number of HH-contacts well.
• Model is a COP (Constraint Optimization Problem)
• Branch and Bound Search for Minimizing Energy
• Combined with Symmetry Breaking
• How good is the propagation?
• Main problem of propagation: bounds on contacts/energy
  From a partial solution, the solver cannot estimate the maximally possible number of HH-contacts well.
Steps

1. Core Construction
2. Mapping
The Advanced Approach: Cubic & FCC

Steps

1. Bounds
2. Core Construction
3. Mapping
Workflow: Predict Best Structure(s) of HP-Sequence

1. \( m = \text{HH-contacts of optimal H-core} \)
2. \( m = m - 1 \)
3. \( \text{Any Solution?} \)
4. \( \text{Next H-core with } m \text{ HH-contacts of size } k \)
5. \( \text{Exists?} \)
6. \( \text{Formulate CSP using H-core} \)
7. \( \text{Solution?} \)
8. \( \text{Optimal structure} \)
9. \( \text{Want all?} \)
10. \( \text{No} \)
Computing Bounds

- Prepares the construction of cores
- How many contacts are possible for $n$ monomers, if freely distributed to lattice points
- Answering the question will give information for core construction
- Main idea: split lattice into layers consider contacts
  - within layers
  - between layers
Layers: Cubic & FCC Lattice
Layers: Cubic & FCC Lattice
Contacts = Layer contacts + Contacts between layers

- Bound Layer contacts: \( \text{Contacts} \leq 2 \cdot n - a - b \)

- Bound Contacts between layers
  - cubic: one neighbor in next layer
    \( \text{Contacts} \leq \min(n_1, n_2) \)
  - FCC: four neighbors in next layer
    \( i - \text{points} \)
Bounding Interlayer Contacts in the FCC

- **Needed:**
  - upper bound for number of contacts between two successive layers in FCC
  - NOTE: Layers only described by parameters \((n_1, a_1, b_1); (n_2, a_2, b_2)\)

- **Method:**
  - compute bounds for number of 1/2/3/4-points of first layer
  - distribute \(n_2\) points greedily
  - technical difficulty: tight bounds of 1/2/3/4-points depend on further parameters

- **Result:** \(B_{\text{FCC}}^{\text{ILC}}(n_1, a_1, b_1, n_2, a_2, b_2)\)

Recall: \(B_{\text{cubic}}^{\text{ILC}}(n_1, a_1, b_1, n_2, a_2, b_2) = \min(n_1, n_2)\)
Recursion Equation for Bounds

- $B_C(n, n_1, a_1, b_1)$: Contacts of core with $n$ elements and first layer $L_1: n_1, a_1, b_1$
- $B_{LC}(n_1, a_1, b_1)$: Contacts in $L_1$
- $B_{ILC}(n_1, a_1, b_1, n_2, a_2, b_2)$: Contacts between $E_1$ and $E_2: n_2, a_2, b_2$
- $B_C(n-n_1, n_2, a_2, b_2)$: Contacts in core with $n - n_1$ elements and first layer $E_2$
Layer sequences

From Recursion:

- by Dynamic Programming: Upper bound on number of contacts
- by Traceback: Set of layer sequences

\[
\text{layer sequence} = (n_1, a_1, b_1), \ldots, (n_4, a_4, b_4)
\]

Set of layer sequences gives distribution of points to layers in all point sets that possibly have maximal number of contacts
Core Construction

Problem

**IN:** number $n$, contacts $c$

**OUT:** all point sets of size $n$ with $c$ contacts

- Optimization problem
- Core construction is a hard combinatorial problem
Problem

**IN:** number $n$, contacts $c$, set of layer sequences $S_{ls}$

**OUT:** all point sets of size $n$ with $c$ contacts and layer sequences in $S_{ls}$

- Use constraints from layer sequences
- Model as constraint satisfaction problem (CSP)

$$(n_1, a_1, b_1), \ldots, (n_4, a_4, b_4) \quad \text{Core} = \text{Set of lattice points}$$
- Number of layers = length of layer sequence
- Number of layers in \( x, y, \) and \( z \): Surrounding Cube
- enumerate numbers of layers \( \Rightarrow \) fix cube \( \Rightarrow \) enumerate points
Workflow

m = HH-contacts of optimal H-core

Next H-core with m HH-contacts of size k

m = m - 1

Any Solution?

Exists?

Formulate CSP using H-core

Solution?

Solve CSP

Optimal structure

Want all?

Yes

No
Mapping Sequences to Cores

find structure such that

- H-Monomers on core positions → hydrophobic core
- all positions differ → self-avoiding
- chain connected → walk

compact core

optimal structure
Given: sequence \( s \) of size \( n \) and \( n_H \) Hs

core \( \text{Core} \) of size \( n_H \)

**CSP Model**

- **Variables** \( X_1, \ldots, X_n \)
  - \( X_i \) is position of monomer \( i \)
- Encode positions as integers

\[
I \left( \begin{array}{c} x \\ y \\ z \end{array} \right) \equiv x + M \times y + M^2 \times z
\]

(unique encoding for 'large enough' \( M \))

- **Constraints**
  1. \( X_i \in \text{Core} \) for all \( s_i = H \)
  2. \( X_i \) and \( X_{i+1} \) are neighbors
  3. \( X_1, \ldots, X_n \) are all different
Constraints for Self-avoiding Walks

- Single Constraints “self-avoiding” and “walk” weaker than their combination
- no efficient algorithm for consistency of combined constraint “self-avoiding walk”
- relaxed combination: stronger and more efficient propagation $k$-avoiding walk constraint

Example: 4-avoiding, but not 5-avoiding
Putting it together

Predict optimal structures by combining the three steps

1. Bounds
2. Core Construction
3. Mapping

Some Remarks

- Pre-compute optimal cores for relevant core sizes
  Given a sequence, only perform Mapping step
- Mapping to cores may fail!
  We use suboptimal cores and iterate mapping.
- Approach extensible to HPNX
  HPNX-optimal structures at least nearly optimal for HP.
- Approach extensible to side chains
  H side chains form core.
Putting it together

Predict optimal structures by combining the three steps

1. Bounds
2. Core Construction
3. Mapping

Some Remarks

- Pre-compute optimal cores for relevant core sizes
  Given a sequence, only perform Mapping step
- Mapping to cores may fail!
  We use suboptimal cores and iterate mapping.
- Approach extensible to HPNX
  HPNX-optimal structures at least nearly optimal for HP.
- Approach extensible to side chains
  H side chains form core.
Putting it together

Predict optimal structures by combining the three steps

1. Bounds
2. Core Construction
3. Mapping

Some Remarks

- Pre-compute optimal cores for relevant core sizes
  Given a sequence, only perform Mapping step

- Mapping to cores may fail!
  We use suboptimal cores and iterate mapping.

- Approach extensible to HPNX
  HPNX-optimal structures at least nearly optimal for HP.

- Approach extensible to side chains
  H side chains form core.
Putting it together

Predict optimal structures by combining the three steps

1. Bounds
2. Core Construction
3. Mapping

Some Remarks

- Pre-compute optimal cores for relevant core sizes
  Given a sequence, only perform Mapping step
- Mapping to cores may fail!
  We use suboptimal cores and iterate mapping.
- Approach extensible to HPNX
  HPNX-optimal structures at least nearly optimal for HP.
- Approach extensible to side chains
  H side chains form core.
Time efficiency

Prediction of one optimal structure
(“Harvard Sequences”, length 48 [Yue et al., 1995])

<table>
<thead>
<tr>
<th>CPSP</th>
<th>PERM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0,1 s</td>
<td>6,9 min</td>
</tr>
<tr>
<td>0,1 s</td>
<td>40,5 min</td>
</tr>
<tr>
<td>4,5 s</td>
<td>100,2 min</td>
</tr>
<tr>
<td>7,3 s</td>
<td>284,0 min</td>
</tr>
<tr>
<td>1,8 s</td>
<td>74,7 min</td>
</tr>
<tr>
<td>1,7 s</td>
<td>59,2 min</td>
</tr>
<tr>
<td>12,1 s</td>
<td>144,7 min</td>
</tr>
<tr>
<td>1,5 s</td>
<td>26,6 min</td>
</tr>
<tr>
<td>0,3 s</td>
<td>1420,0 min</td>
</tr>
<tr>
<td>0,1 s</td>
<td>18,3 min</td>
</tr>
</tbody>
</table>

- **CPSP**: “our approach”, constraint-based
- **PERM** [Bastolla et al., 1998]: stochastic optimization
Many Optimal Structures

Sequence HPPHPPPPHP

- There can be many ...
- HP-model is degenerated
- Number of optimal structures = degeneracy
## Completeness

Predicted number of all optimal structures ("Harvard Sequences")

<table>
<thead>
<tr>
<th>CPSP</th>
<th>CHCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.677.113</td>
<td>1500 \times 10^3</td>
</tr>
<tr>
<td>28.180</td>
<td>14 \times 10^3</td>
</tr>
<tr>
<td>5.090</td>
<td>5 \times 10^3</td>
</tr>
<tr>
<td>1.954.172</td>
<td>54 \times 10^3</td>
</tr>
<tr>
<td>1.868.150</td>
<td>52 \times 10^3</td>
</tr>
<tr>
<td>106.582</td>
<td>59 \times 10^3</td>
</tr>
<tr>
<td>15.926.554</td>
<td>306 \times 10^3</td>
</tr>
<tr>
<td>2.614</td>
<td>1 \times 10^3</td>
</tr>
<tr>
<td>580.751</td>
<td>188 \times 10^3</td>
</tr>
</tbody>
</table>

- **CPSP**: “our approach”
- **CHCC** [Yue *et al.*, 1995]: complete search with hydrophobic cores
Unique Folder

- HP-model degenerated
- Low degeneracy $\approx$ stable $\approx$ protein-like
- Are there protein-like, unique folder in 3D HP models?
- How to find out?
Unique Folder

- HP-model degenerated
- Low degeneracy $\approx$ stable $\approx$ protein-like
- Are there protein-like, unique folder in 3D HP models?
- How to find out?

MC-search through sequence space
Unique Folder

- HP-model degenerated
- Low degeneracy \( \approx \) stable \( \approx \) protein-like
- Are there protein-like, unique folder in 3D HP models?
- How to find out?

Yes: many, e.g. about 10,000 for \( n=27 \)
Software: CPSP Tools

http://cpsp.informatik.uni-freiburg.de

CPSP Tools

Constraint-based Protein Structure Prediction

Bioinformatics Group
Albert-Ludwigs-University Freiburg

web-tools version 1.1.1 (06.04.2011)

The CPSP-tools package provides programs to solve exactly and completely the problems typical of studies using 3D lattice protein models. Among the tasks addressed are the prediction of globally optimal and/or suboptimal structures as well as sequence design and neutral network exploration.

Choose a tool from the left for ad hoc usage
(CPSP-tools version 2.4.2) (LatPack version 1.7.2)

or

Download the full CPSP-tools or LatPack package for local usage!

If you use the CPSP-tools please cite the following publications:

- Martin Mann, Sebastian Will, and Rolf Backofen.
  CPSP-tools - Exact and Complete Algorithms for High-throughput 3D Lattice Protein Studies.
  In BMC Bioinformatics, 9, 230, 2008.