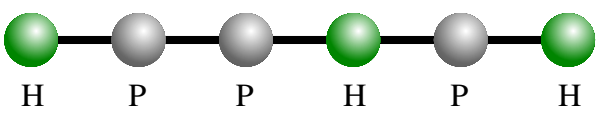
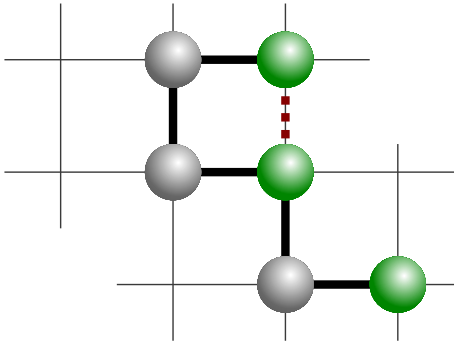
Lattice Models: The Simplest Protein Model

The HP-Model (Lau & Dill, 1989)

- model only hydrophobic interaction
  - alphabet $\{H, P\}$; H/P = hydrophobic/polar
  - energy function favors HH-contacts
- structures are discrete, simple, and originally 2D
  - model only backbone (C-α) positions
  - structures are drawn (originally) on a square lattice $\mathbb{Z}^2$ without overlaps: Self-Avoiding Walk

Example





HH-contact

HP-Model Definition

Definition

The HP-model is a protein model, where

- Sequence $s \in \{H, P\}^n$
- Structure $\omega : [1..n] \to L$ (e.g. $L = \mathbb{Z}^2, L = \mathbb{Z}^3$), s.t.
  1. for all $1 \leq i < n$:
     $d(\omega(i), \omega(i+1)) = d_{\min}(L) \qquad [d_{\min}(\mathbb{Z}^2) = 1]$
  2. for all $1 \leq i < j \leq n$: $\omega(i) \neq \omega(j)$
- Energy function $\mathrm{E}(s, \omega) = \sum_{1 \leq i < j \leq n} E_{s_i, s_j} \Delta(\omega(i), \omega(j))$,

where $E =$

	H	P
H	-1	0
P	0	0

and $\Delta(p, q) = \begin{cases} 1 & \text{if } d(p, q) = d_{\min}(L) \\ 0 & \textit{otherwise} \end{cases}$

HP-Model Definition

Definition

The HP-model is a protein model, where

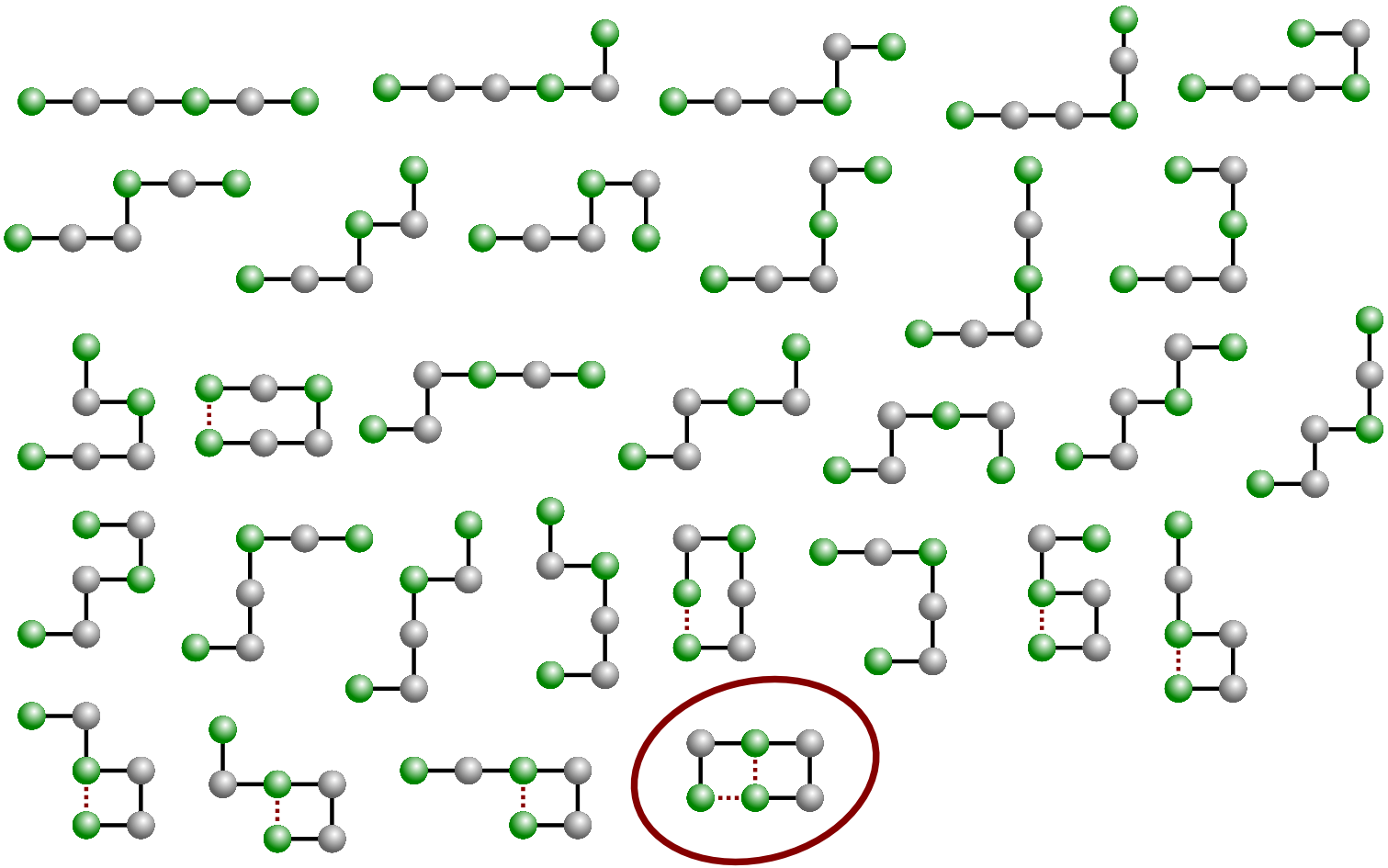
- Sequence $s \in \{H, P\}^n$
- Structure $\omega : [1..n] \to L$ (e.g. $L = \mathbb{Z}^2, L = \mathbb{Z}^3$), s.t.
  1. for all $1 \leq i < n$:
     $d(\omega(i), \omega(i+1)) = d_{\min}(L) \qquad [d_{\min}(\mathbb{Z}^2) = 1]$
  2. for all $1 \leq i < j \leq n$: $\omega(i) \neq \omega(j)$
- Energy function $\mathrm{E}(s, \omega) = \sum_{1 \leq i < j \leq n} E_{s_i, s_j} \Delta(\omega(i), \omega(j))$,

$$\text{where } E = \begin{array}{c|cc} & H & P \\ \hline H & -1 & 0 \\ P & 0 & 0 \end{array}$$

$$\text{and } \Delta(p, q) = \begin{cases} 1 & \text{if } d(p, q) = d_{\min}(L) \\ 0 & \textit{otherwise} \end{cases}$$

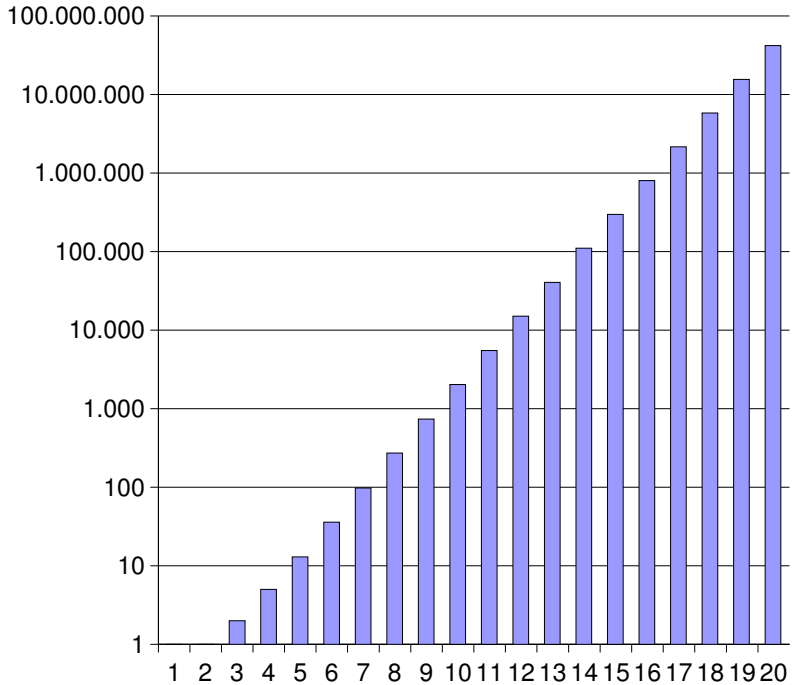
Structures in the HP-Model

Sequence HPPHPH



How many structures are there?

Self-avoiding Walks of the Square Lattice (without Symmetry)



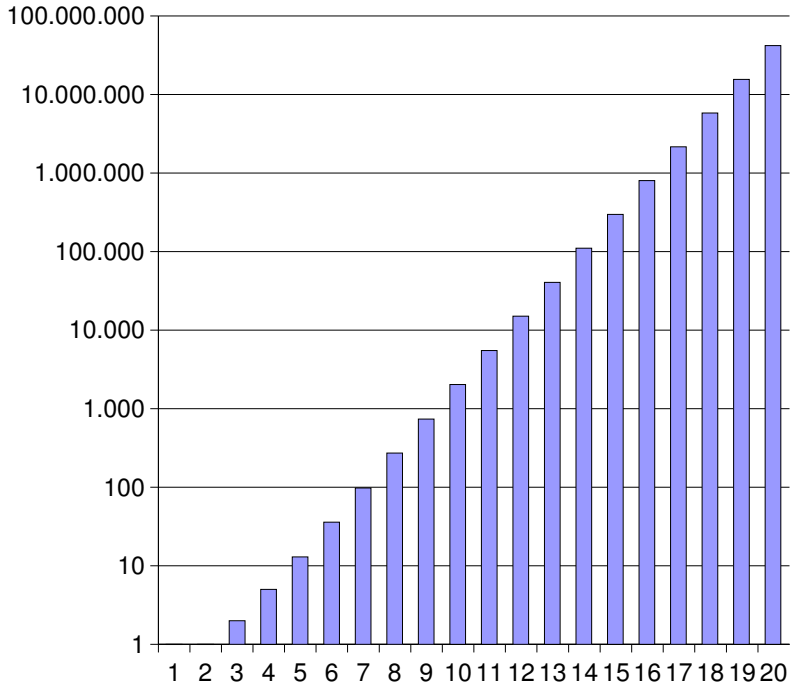
Naive enumeration not possible. Even NP-complete:

B. Berger, T. Leighton. Protein folding in the hydrophobic-hydrophilic (HP) Model is NP-complete. RECOMB'98

P. Crescenzi. D. Goldman. C. Paoadimitriou. A. Piccolbom, and M. Yakakis. On the complexity of protein folding. RECOMB'98

How many structures are there?

Self-avoiding Walks of the Square Lattice (without Symmetry)



Naive enumeration not possible. Even NP-complete:

B. Berger, T. Leighton. Protein folding in the hydrophobic-hydrophilic (HP) Model is NP-complete. RECOMB'98

P. Crescenzi. D. Goldman. C. Paoadimitriou. A. Piccolbom, and M. Yakakis. On the complexity of protein folding. RECOMB'98

Constraint Programming (CP)

- Model and solve *hard combinatorial problems* as CSP by search and propagation
- cf. ILP, but CP offers more flexible modeling and differs in solving strategies

Definition

A Constraint Satisfaction Problems (CSP) consists of

- *variables* $\mathcal{X} = \{X_1, \ldots, X_n\}$,
- the domain D that associates finite domains $D_1 = D(X_1), \ldots, D_n = D(X_n)$ to $\mathcal{X}$.
- a set of constraints C.

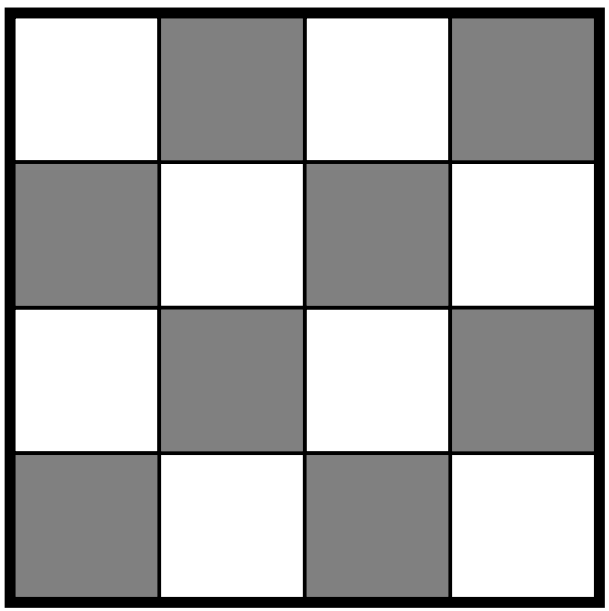
A solution is an assignment of variables to values of their domains that satisfies the constraints.

Commercial Impact of Constraints Programming

Michelin and Dassault, Renault	Production planning
Lufthansa, Swiss Air, ...	Staff planning
Nokia	Software configuration
Siemens	Circuit verification
French National Railway Company	Train schedule
...	...

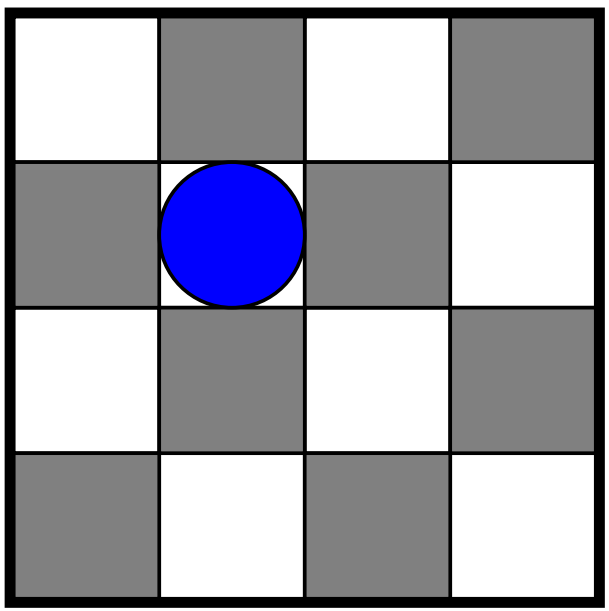
CP Example: The N-Queens Problem

4-Queens: place 4 queens on 4×4 board without attacks



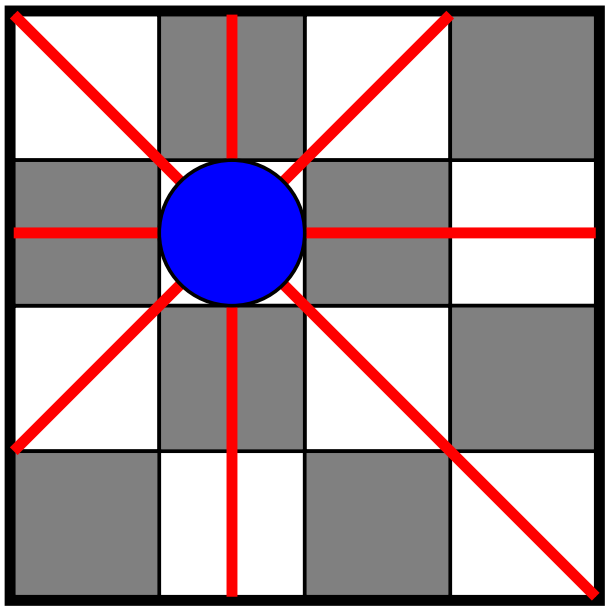
CP Example: The N-Queens Problem

4-Queens: place 4 queens on 4×4 board without attacks



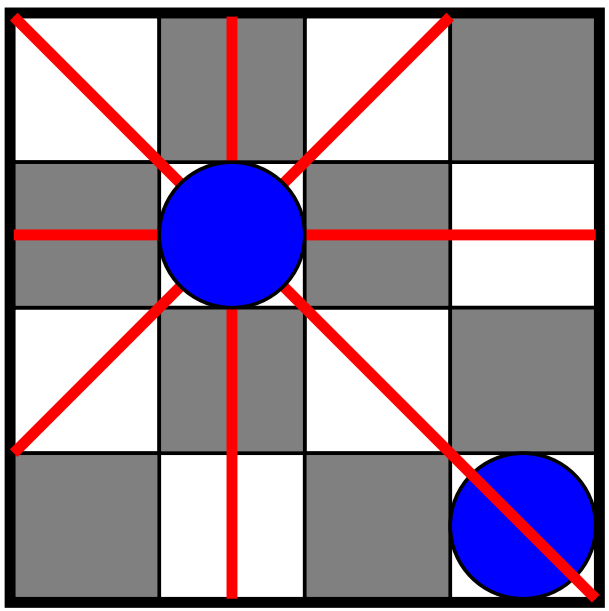
CP Example: The N-Queens Problem

4-Queens: place 4 queens on 4×4 board without attacks



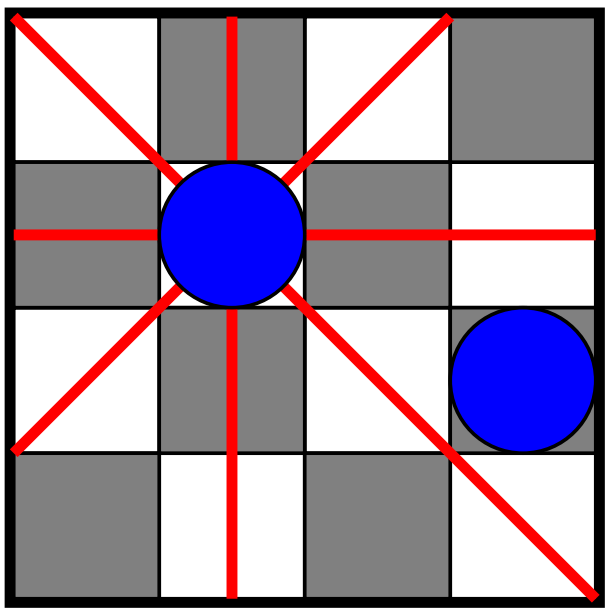
CP Example: The N-Queens Problem

4-Queens: place 4 queens on 4×4 board without attacks



CP Example: The N-Queens Problem

4-Queens: place 4 queens on 4×4 board without attacks



Model 4-Queens as CSP (Constraint Model)

- Variables

$$X_1, \ldots, X_4$$

$X_i = j$ means "queen in column i, row j"

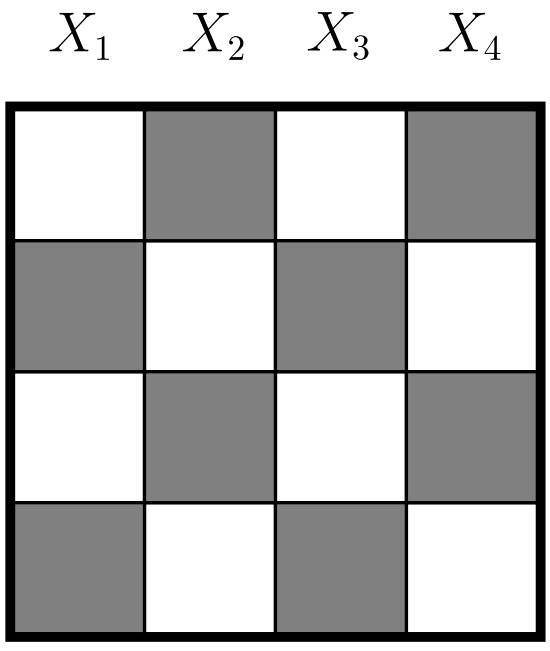
- Domains

$$D(X_i) = \{1, \ldots, 4\} \text{ for } i = 1..4$$

- Constraints (for $i, i' = 1..4$ and $i \neq i'$)

$$\begin{aligned} X_i &\neq X_{i'} && \text{(no horizontal attack)} \\ i - X_i &\neq i' - X_{i'} && \text{(no attack in first diagonal)} \\ i + X_i &\neq i' + X_{i'} && \text{(no attack in second diagonal)} \end{aligned}$$

Solving 4-Queens by Search and Propagation, $X_1 = 1$

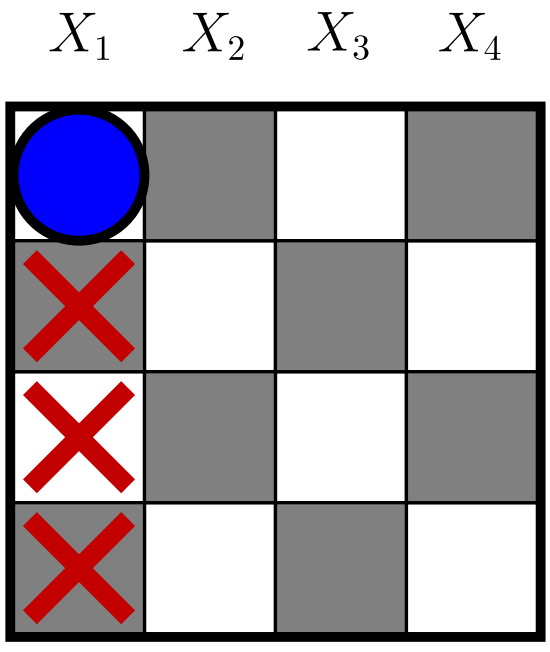


$$X_1, \ldots, X_4$$

$$D(X_i) = \{1, \ldots, 4\} \text{ for } i = 1..4$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 1$

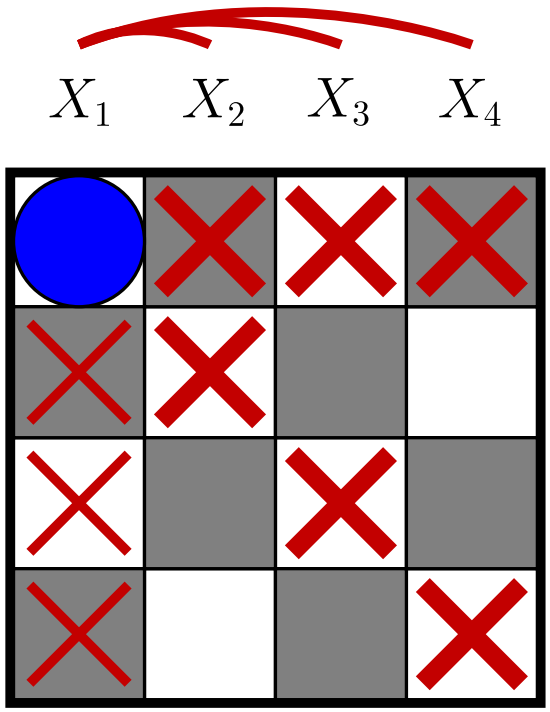


$$X_1, \ldots, X_4$$

$$D(X_1) = \{1\}, D(X_i) = \{1, \ldots, 4\} \text{ for } i = 2..4$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 1$

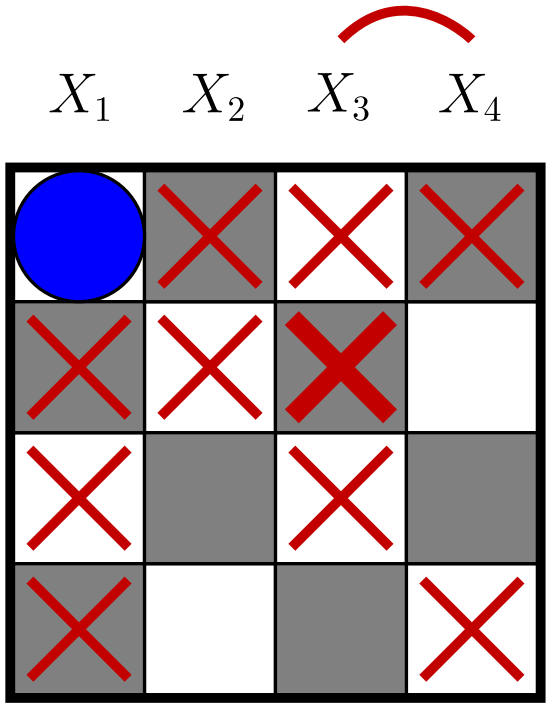


$$X_1, \ldots, X_4$$

$$D(X_1) = \{1\}, D(X_2) = \{3, 4\}, D(X_3) = \{2, 4\}, D(X_4) = \{2, 3\}$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 1$

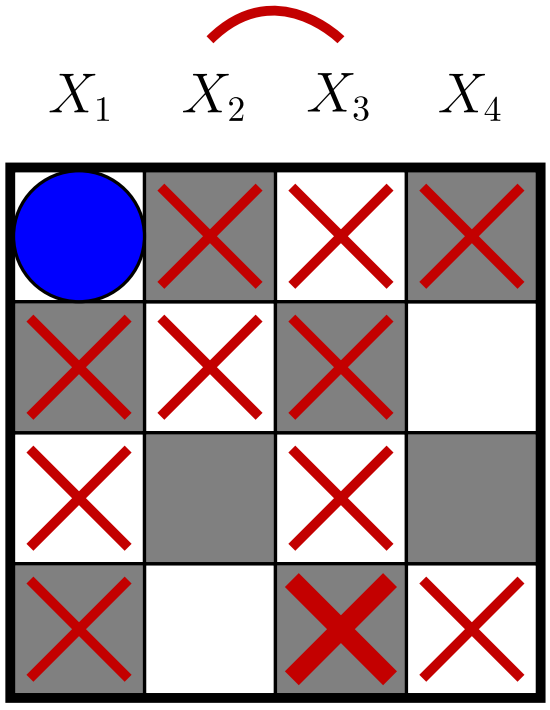


$$X_1, \ldots, X_4$$

$$D(X_1) = \{1\}, D(X_2) = \{3, 4\}, D(X_3) = \{4\}, D(X_4) = \{2, 3\}$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 1$

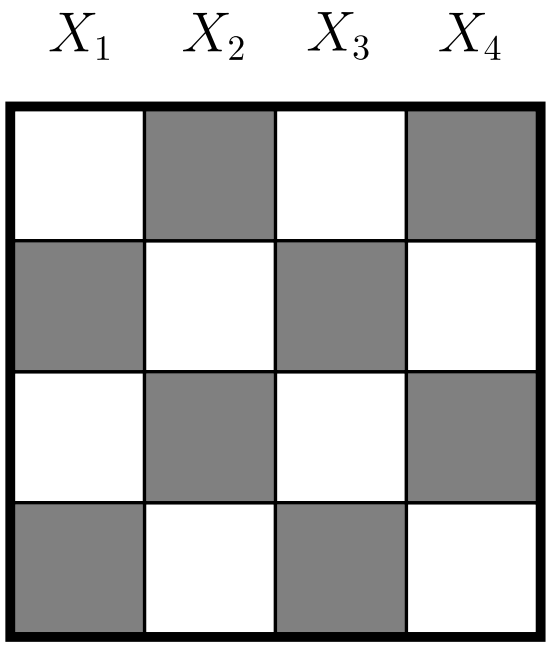


$$X_1, \ldots, X_4$$

$$D(X_1) = \{1\}, D(X_2) = \{3, 4\}, D(X_3) = \{\}, D(X_4) = \{2, 3\}$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 2$

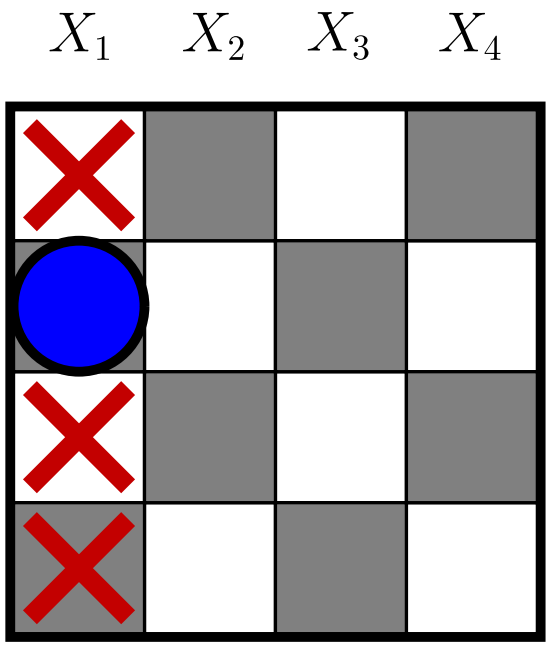


$$X_1, \ldots, X_4$$

$$D(X_i) = \{1, \ldots, 4\} \text{ for } i = 1..4$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 2$

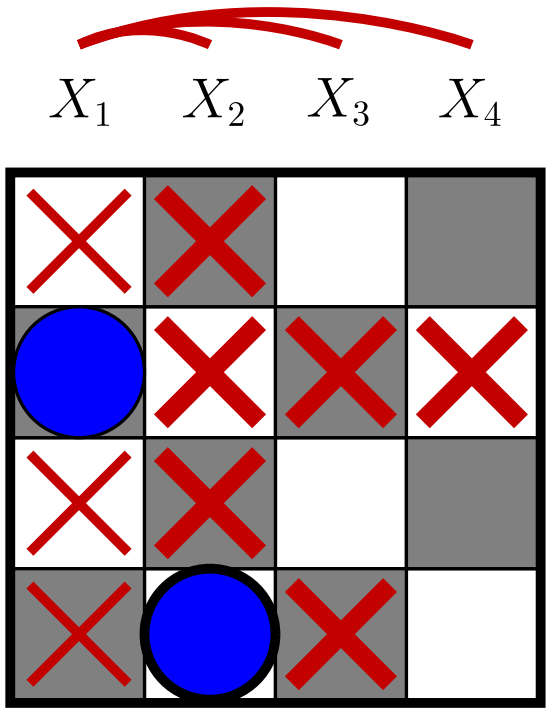


$$X_1, \ldots, X_4$$

$$D(X_1) = \{2\}, D(X_i) = \{1, \ldots, 4\} \text{ for } i = 2..4$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 2$

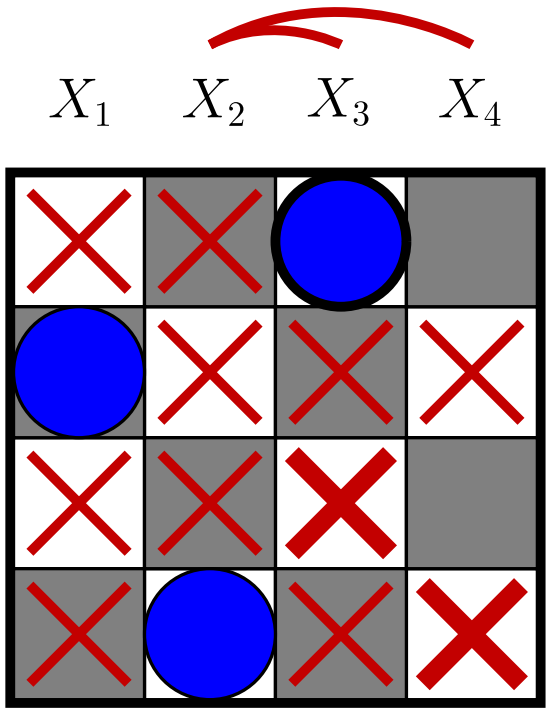


$$X_1, \ldots, X_4$$

$$D(X_1) = \{2\}, D(X_2) = \{4\}, D(X_3) = \{1, 3\}, D(X_4) = \{1, 3, 4\}$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 2$

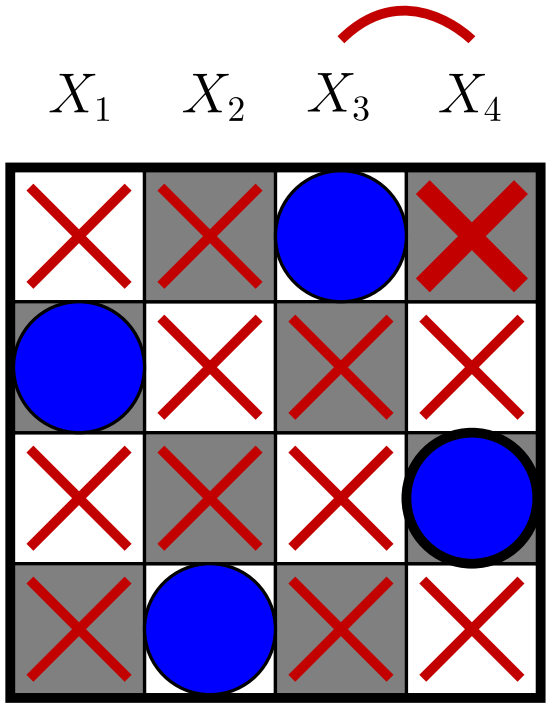


$$X_1, \ldots, X_4$$

$$D(X_1) = \{2\}, D(X_2) = \{4\}, D(X_3) = \{1\}, D(X_4) = \{3, 4\}$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Solving 4-Queens by Search and Propagation, $X_1 = 2$



$$X_1, \ldots, X_4$$

$$D(X_1) = \{2\}, D(X_2) = \{4\}, D(X_3) = \{1\}, D(X_4) = \{3\}$$

$$X_i \neq X_{i'}, i - X_i \neq i' - X_{i'}, i + X_i \neq i' + X_{i'}$$

Constraint Optimization

Definition

A Constraint Optimization Problem (COP) is a CSP together with an objective function f on solutions.
A solution of the COP is a solution of the CSP that maximizes/minimizes f.

Solving by Branch & Bound Search
Idea of B&B:

- Backtrack & Propagate as for solving the CSP
- Whenever a solution s is found, add constraint "next solutions must be better than $f(s)$".

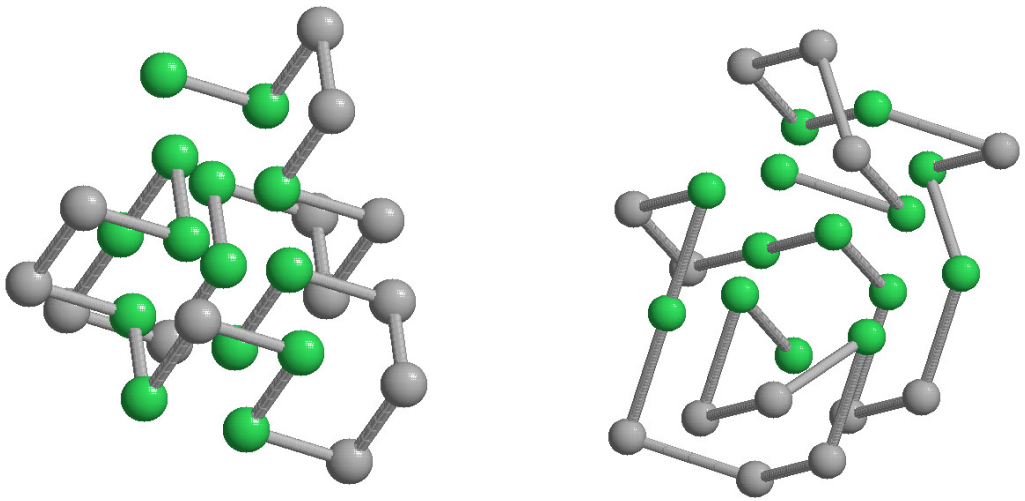
Exact Prediction in 3D cubic & FCC

The problem

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

HHPPPHHPHHPPHHHPPHHPPPHPPHH

OUT: self avoiding walk ω on cubic/fcc lattice with minimal HP-energy $E_{\mathrm{HP}}(s, \omega)$



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 ith monomer $\omega(i)$

- Domains

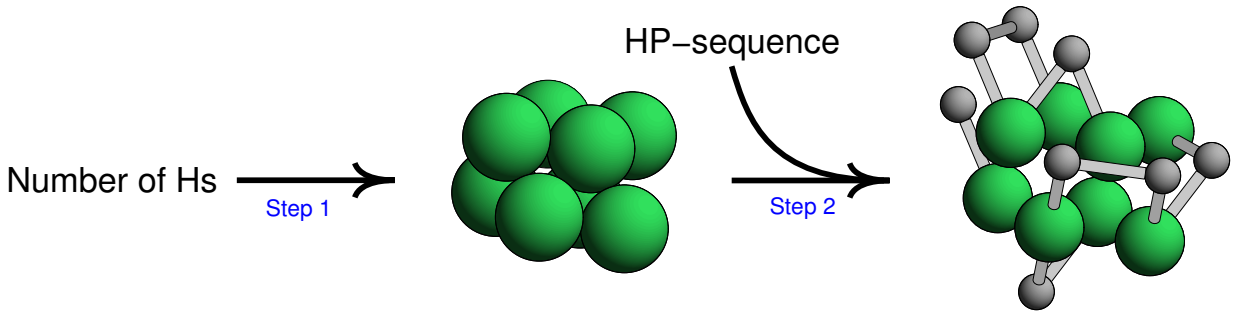
$$D(X_i) = D(Y_i) = D(Z_i) = \{-n, \ldots, n\}$$

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

Solving the First Model

- Model is a COP (Constraint Optimization Problem)
- Branch and Bound Search for Minimizing *Energy*
- (Add 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.

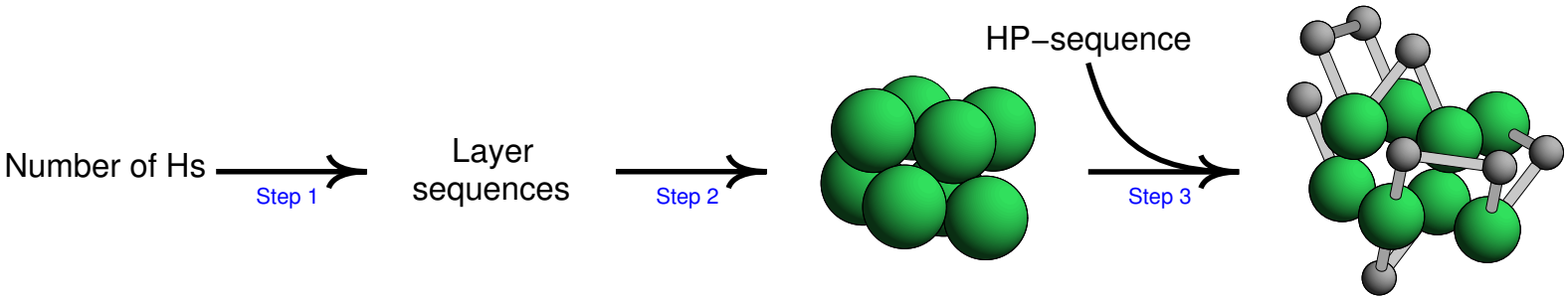
The Advanced Approach: Cubic & FCC



Steps

1. Core Construction
2. Mapping

The Advanced Approach: Cubic & FCC



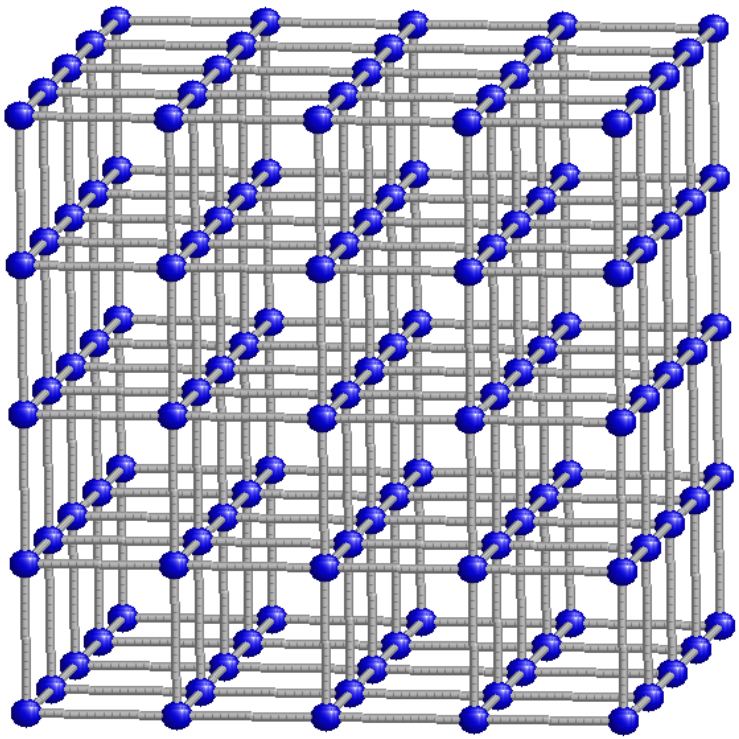
Steps

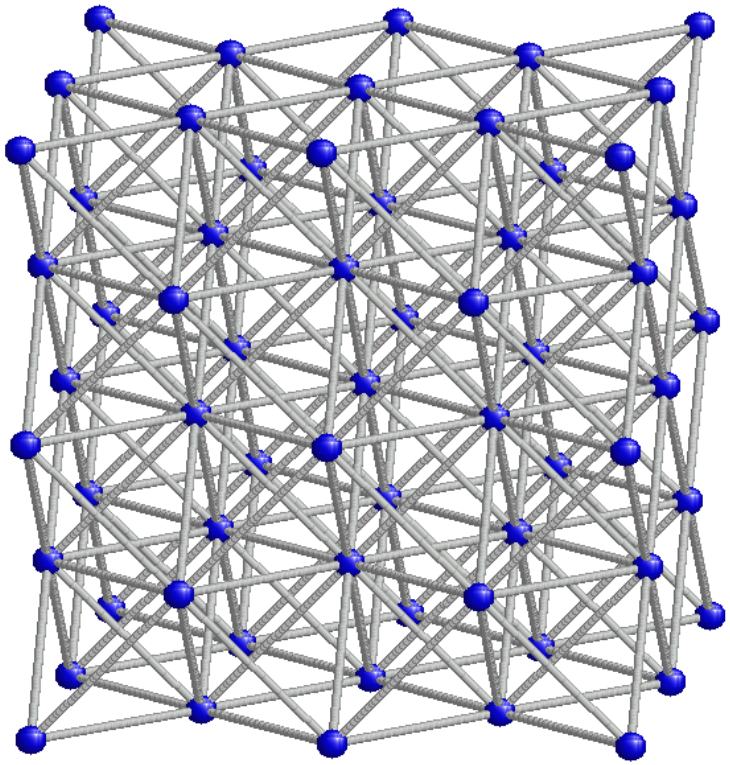
1. Bounds
2. Core Construction
3. Mapping

Computing Bounds

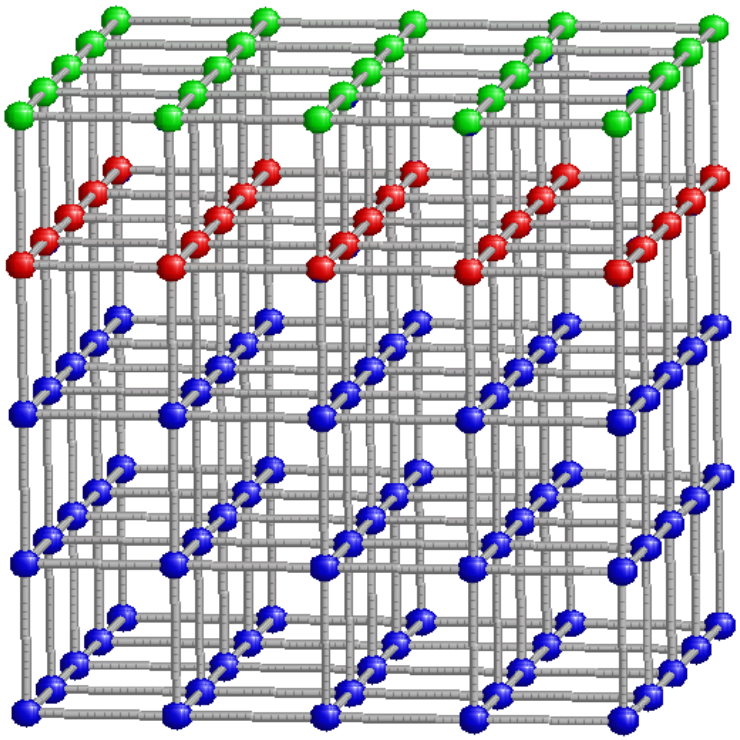
- Prepares the construction of cores
- How many contacts are possible for n monomers, if freely distributed to lattice points
- Answering the question will give information for core construction
- Main idea: split lattice into layers
  consider contacts
  - within layers
  - between layers

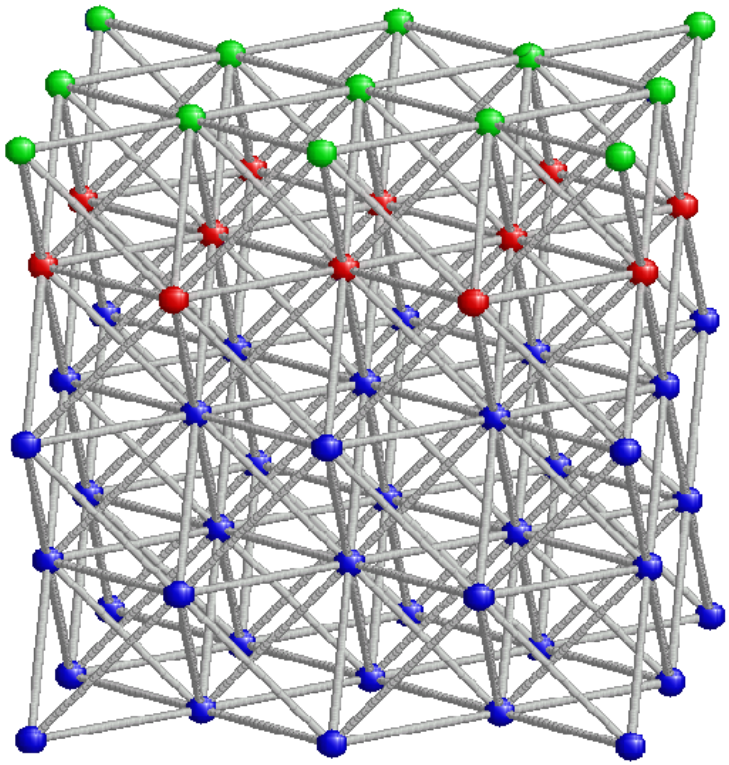
Layers: Cubic & FCC Lattice





Layers: Cubic & FCC Lattice



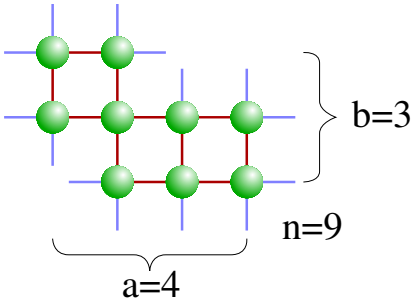


Contacts

Contacts =

Layer contacts + Contacts between layers

- Bound Layer contacts: Contacts $\leq 2 \cdot n - a - b$

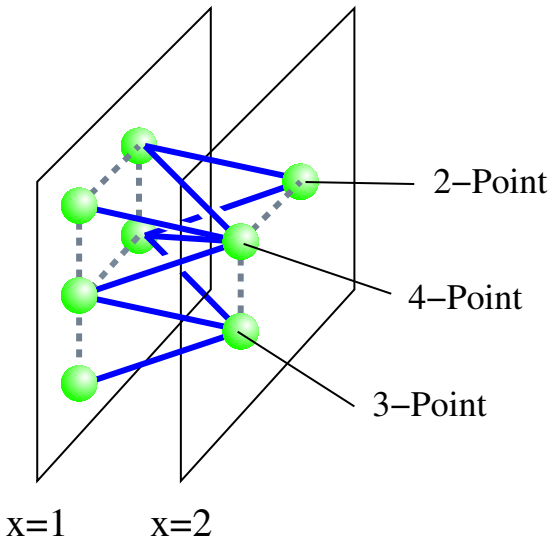


- Bound Contacts between layers
  - cubic: one neighbor in next layer

    $Contacts \leq \min(n_1, n_2)$

  - FCC: **four** neighbors in next layer

    i – points

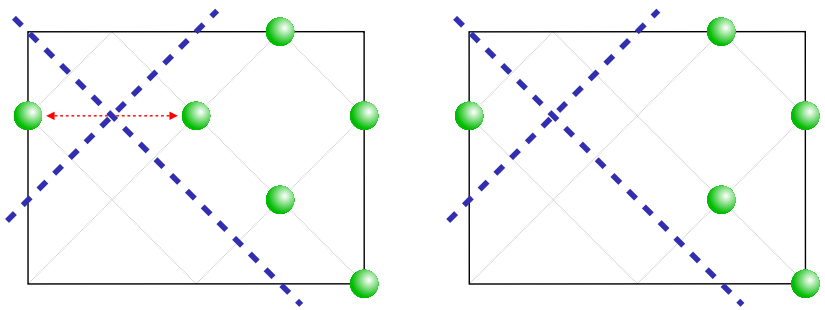


i-points

Layer L_1 : $n_1, a_1, b_1, m_{\mathrm{nc}1}, m_{\mathrm{nt}1}, m_{\mathrm{x}1}$

Number of i-points $\#i$ in L_1

$$\#4 = n_1 - a_1 - b_1 + 1 + m_{\mathrm{nc}1}$$
$$\#3 = m_{\mathrm{x}1} - 2(m_{\mathrm{nc}1} - m_{\mathrm{nt}1})$$
$$\#2 = 2a_1 + 2b_1 - 4 - 2\#3 - 3m_{\mathrm{nc}1} - m_{\mathrm{nt}1}$$
$$\#1 = \#3 + 2m_{\mathrm{nc}1} + 2m_{\mathrm{nt}1} + 4$$



Contacts between Layers

Layer $L_1 : n_1, a_1, b_1, m_{\text{nc}1}, m_{\text{nt}1}, m_{\text{x}1}$, Layer $L_2 : n_2$

Theorem (Number of contacts between layers)

(Eliminate parameter $m_{\text{x}1}$)

$$\#3' = \text{maximal number 3-points for } n_1, a_1, b_1, m_{\text{nc}1}, m_{\text{nt}1}$$

$$\hookrightarrow \quad \#2' = 2a_1 + 2b_1 - 4 - 2\#3' - 4m_{\text{nc}1}$$

$$\#1' = \#3' + 4m_{\text{nc}1} + 4 \quad \#4' = \#4$$

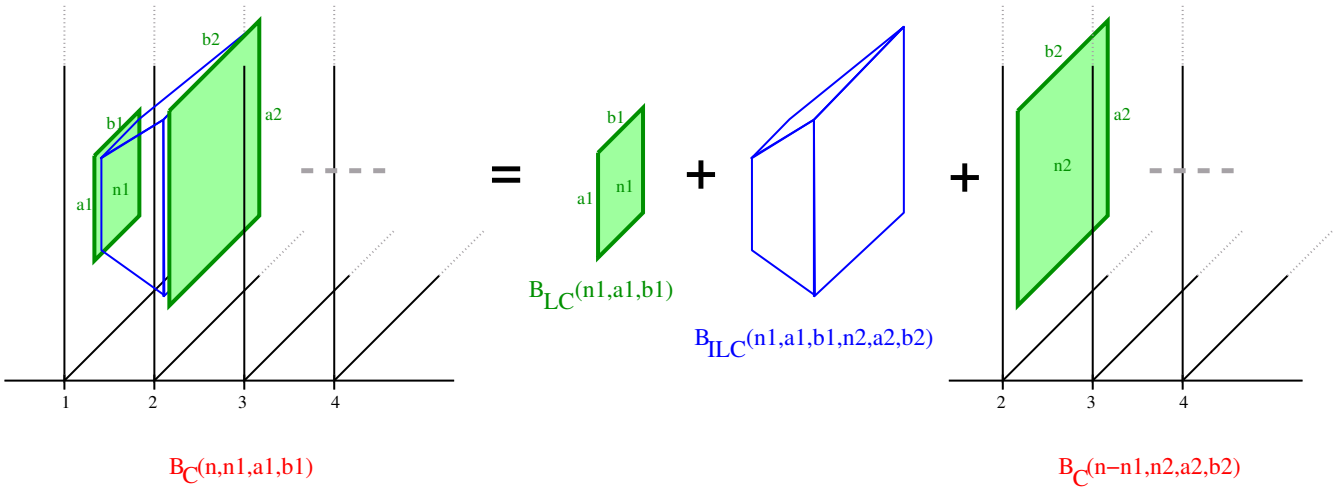
(Distribute n' points optimally to i-points in L_1)

$$b_4 = \min(n_2, \#4') \qquad b_3 = \min(n_2 - b_4, \#3')$$

$$b_2 = \min(n_2 - b_4 - b_3, \#2') \qquad b_1 = \min(n_2 - b_4 - b_3 - b_2, \#1')$$

$$\boxed{\text{Contacts between } L_1 \text{ and } L_2 \quad \leq \quad 4 \cdot b_4 + 3 \cdot b_3 + 2 \cdot b_2 + b_1}$$

Recursion Equation for Bounds

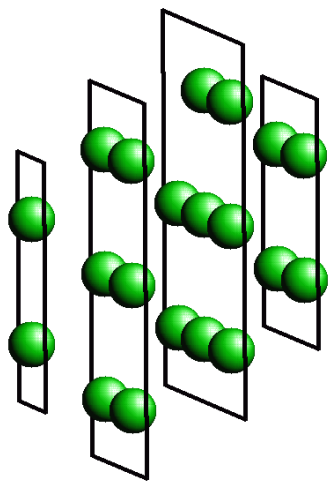


- $\mathrm{B_C}(n, n_1, a_1, b_1)$: Contacts of core with n elements and first layer L_1 : n_1, a_1, b_1
- $\mathrm{B_{LC}}(n_1, a_1, b_1)$: Contacts in L_1
- $\mathrm{B_{ILC}}(n_1, a_1, b_1, n_2, a_2, b_2)$: Contacts between E_1 and E_2 : n_2, a_2, b_2
- $\mathrm{B_C}(n - n_1, n_2, a_2, b_2)$: Contacts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



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

Poblem

IN: number n, contacts c

OUT: all point sets of size n with c contacts

- Optimization problem
- Core construction is a hard combinatorial problem

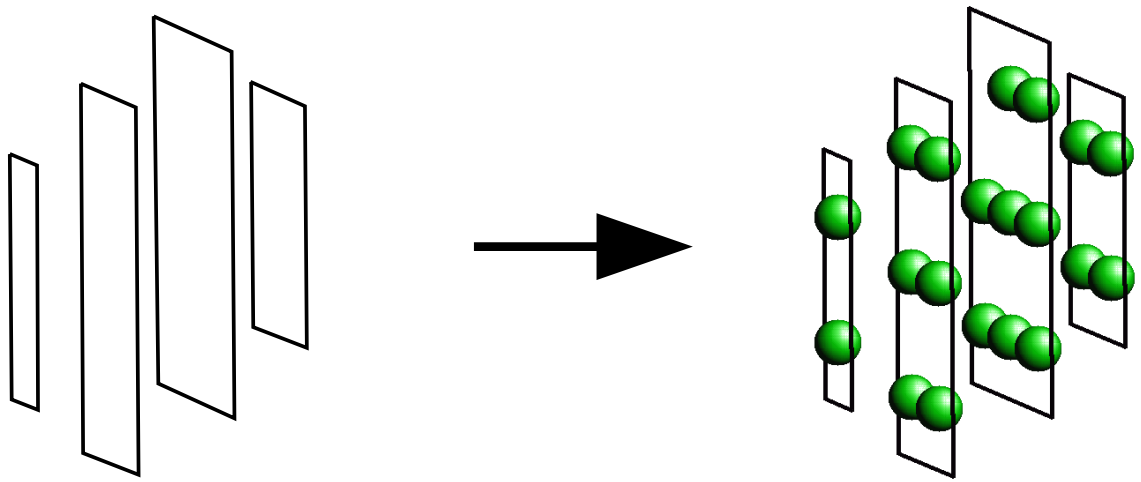
Core construction: Modified Problem

Poblem

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

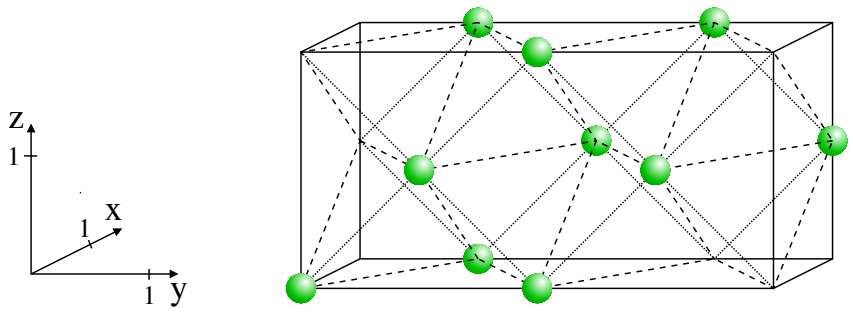
OUT: all point sets of size n with c contacts and layer sequences in S_{ls}

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



$(n_1, a_1, b_1), \ldots, (n_4, a_4, b_4)$ Core = Set of lattice points

Core Construction — Details

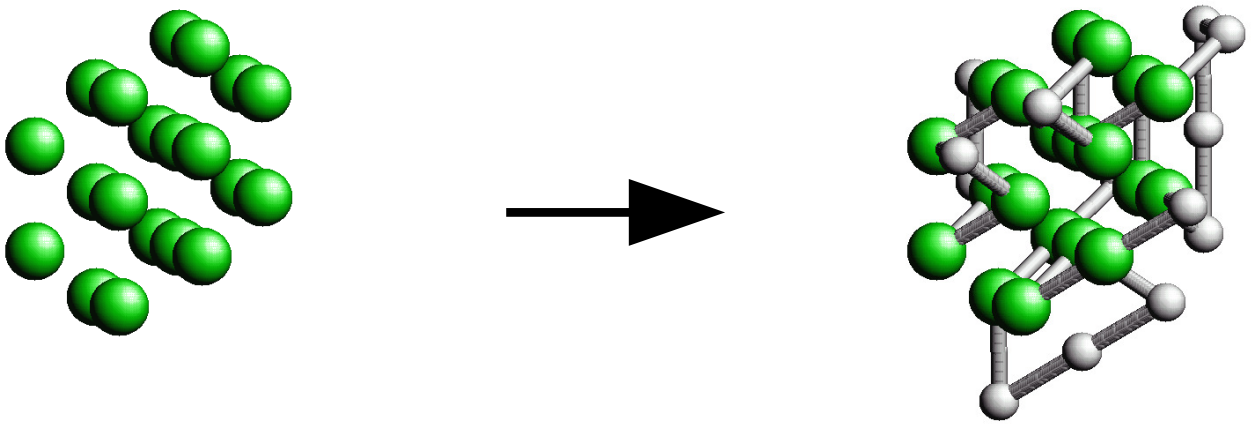


- Number of layers = length of layer sequence
- Number of layers in x, y, and z: Surrounding Cube
- enumerate layers $\Rightarrow$ fix cube $\Rightarrow$ enumerate points

Mapping Sequences to Cores

find structure such that

- H-Monomers on core positions $\rightarrow$ hydrophobic core
- all positions differ $\rightarrow$ self-avoiding
- chain connected $\rightarrow$ walk



compact core optimal structure

Mapping Sequence to Cores — CSP

Given: sequence s of size n and n_H Hs
core *Core* of size n_H

CSP Model

- Variables $X_1, \ldots, X_n$
  X_i is position of monomer i

  Encode positions as integers

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \equiv M^2 * x + M * y + z$$

  (unique encoding for 'large enough' M)

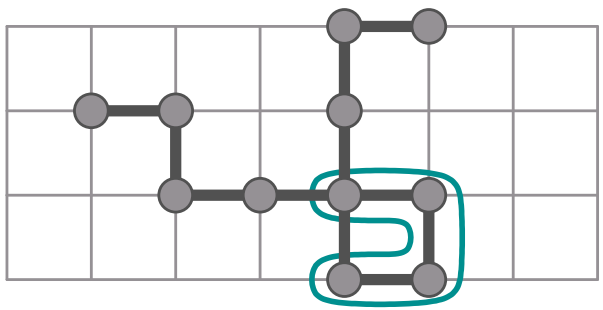
- Constraints
  1. $X_i \in \mathit{Core}$ for all $s_i = H$
  2. X_i and X_{i+1} are neighbors
  3. $X_1, \ldots, X_n$ are all different

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.

Time efficiency

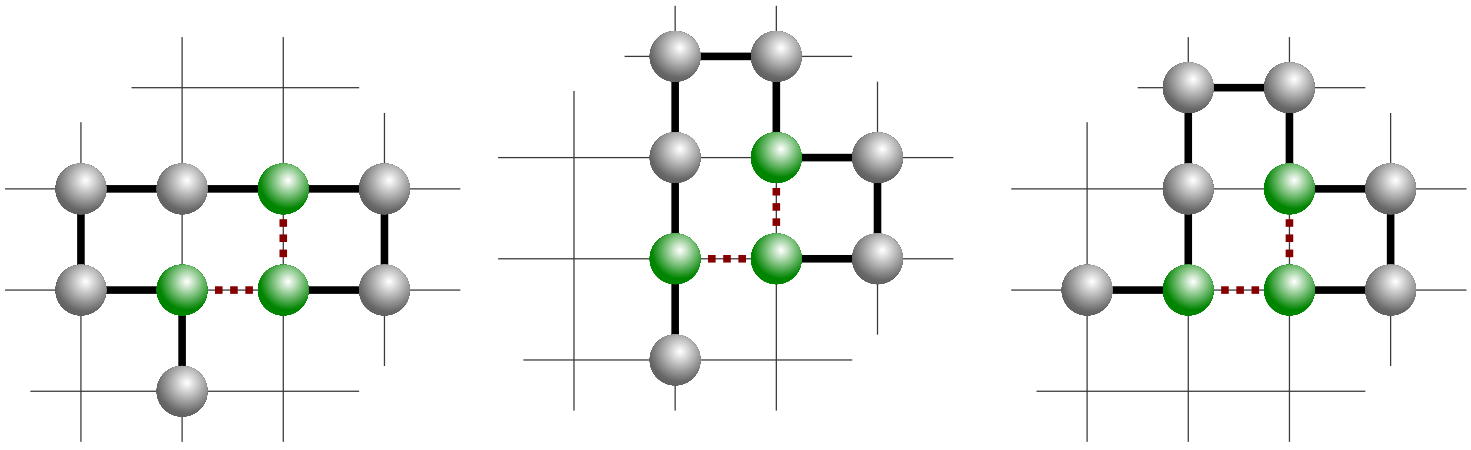
Prediction of one optimal structure
("Harvard Sequences", length 48 [Yue *et al.*, 1995])

CPSP	PERM
0,1 s	6,9 min
0,1 s	40,5 min
4,5 s	100,2 min
7,3 s	284,0 min
1,8 s	74,7 min
1,7 s	59,2 min
12,1 s	144,7 min
1,5 s	26,6 min
0,3 s	1420,0 min
0,1 s	18,3 min

- CPSP: "our approach", constraint-based
- PERM [Bastolla *et al.*, 1998]: stochastic optimization

Many Optimal Structures

Sequence HPPHPPPHP



. . . ?

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

Completeness

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

CPSP	CHCC
10.677.113	1500×10^3
28.180	14×10^3
5.090	5×10^3
1.954.172	54×10^3
1.868.150	52×10^3
106.582	59×10^3
15.926.554	306×10^3
2.614	1×10^3
580.751	188×10^3

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

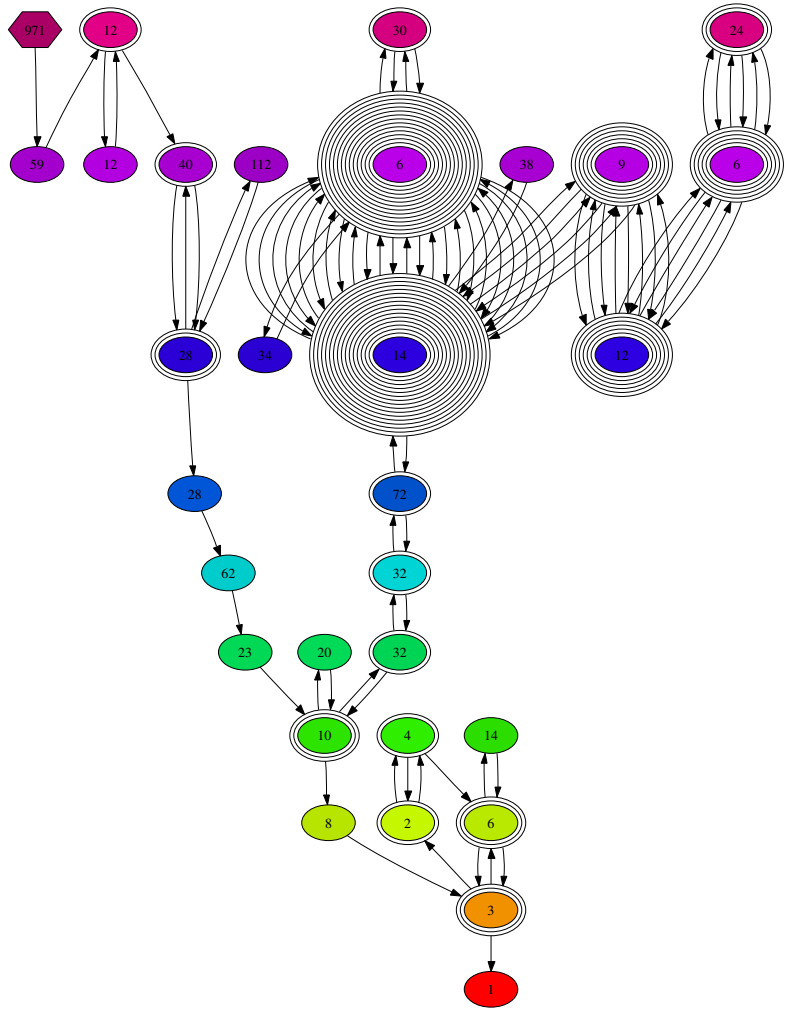
Unique Folder

- 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?

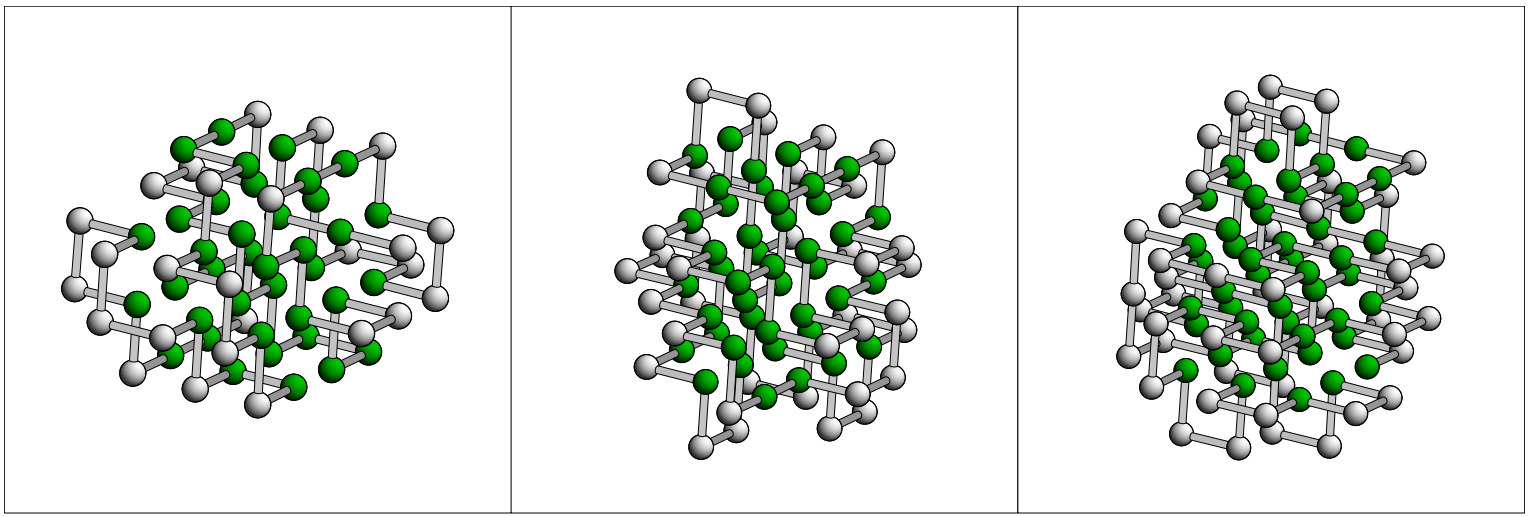
MC-search through sequence space



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:8080/index.jsp

Menu

Home

HPstruct
structure pred.

HPconvert
PDB, CML, ...

HPview
3D visualization

HPdeg
degeneracy

HPnnet
neutral network

HPdesign
seq. design

LatFit
PDB to lattice

Results
direct access

Help

FAQ

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.