# A Constraint-Based Approach to Structure Prediction for Simplified Protein Models that Outperforms Other Existing Methods

**Rolf Backofen and Sebastian Will**

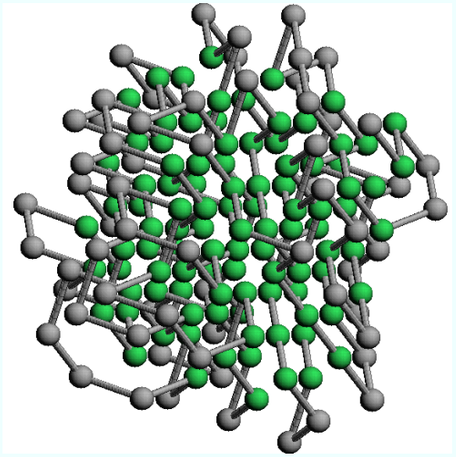
**Chair for Bioinformatics**

**Friedrich-Schiller-Universität Jena**

**new affiliation: Albert-Ludwigs-Universität Freiburg**

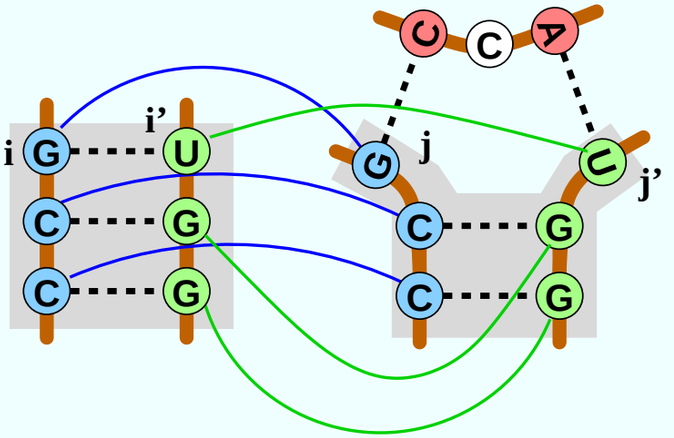
# Our Group/Research Interest

- protein folding in simplified models



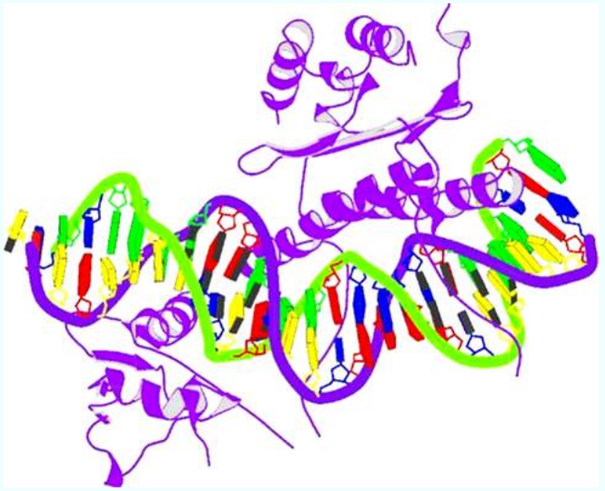
  - Sebastian Will

- RNA-sequence-structure alignment



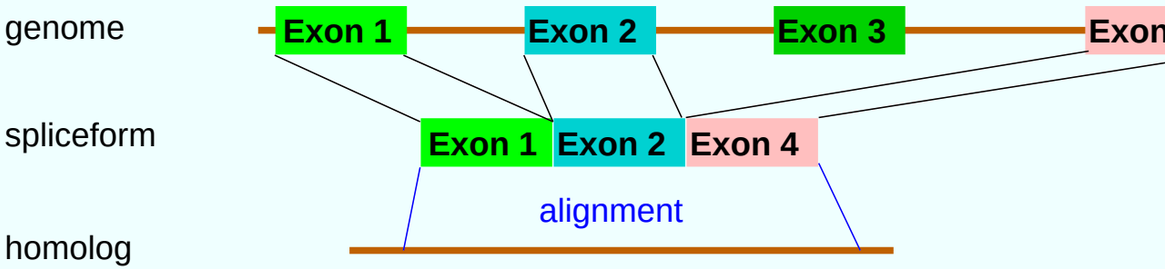

  - Sven Siebert, Sebastian Will

- recognition of regulatory sequences



  - Rainer Pudimat (JCB)

- alternative splicing




  - Michael Hiller

- selenoproteins: Anke Busch, Sven Siebert (DFG-Schwerpunkt “Selenoproteine”)

# Overview

**computer scientist are interested in methods**

- **method: constraint-based structure prediction**
  - **lattice models**
  - **basic model of HP-type models**
  - **subproblems: bounds, hydropbic cores, threading**

# Overview

**computer scientist are interested in methods**

- **method: constraint-based structure prediction**
  - **lattice models**
  - **basic model of HP-type models**
  - **subproblems: bounds, hydropbic cores, threading**

**bioinformatics are interested in applications as well**

- **results and applications**
  - **degeneracy of sequences**
  - **finding protein-likes sequences with unique ground state**
  - **comparing different models (cubic/fcc, HP-model with HPNX)**

# Structure Prediction as Optimization Problem

- searched: structure (conformation) of minimal (free) energy

  $\Rightarrow$ huge search space

- hence: only parts of the search space considered $\Rightarrow$ generate-and-test
  - generate approximation
  - here: broad exploration of search space
  - starting points for fine-tuning

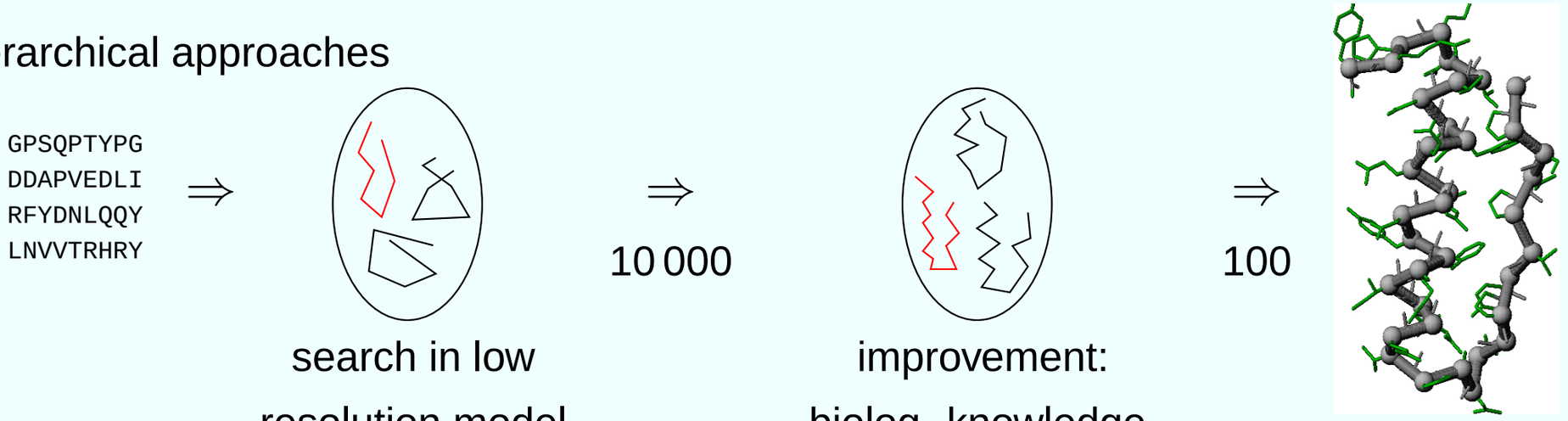
# Structure Prediction as Optimization Problem

- searched: structure (conformation) of minimal (free) energy

  $\Rightarrow$ huge search space

- hence: only parts of the search space considered $\Rightarrow$ generate-and-test
  - generate approximation
  - here: broad exploration of search space
  - starting points for fine-tuning

- hierarchical approaches




search in low resolution model

improvement: biolog. knowledge, molecular dynamics

# Structure Prediction as Optimization Problem

- searched: structure (conformation) of minimal (free) energy

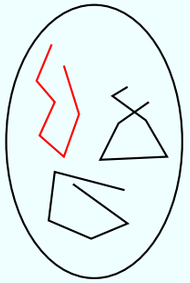
  $\Rightarrow$ huge search space

- hence: only parts of the search space considered $\Rightarrow$ generate-and-test
  - generate approximation
  - here: broad exploration of search space
  - starting points for fine-tuning

- hierarchical approaches

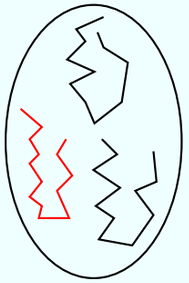
GPSQPTYPG
DDAPVEDLI
RFYDNLQQY
LNVVTRHRY

$\Rightarrow$



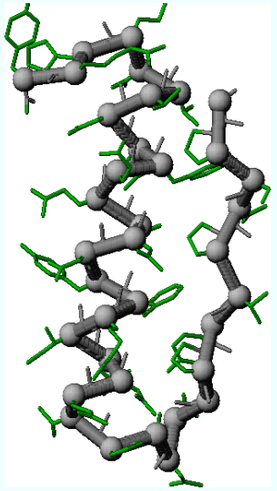
search in low resolution model

$\Rightarrow$ 10 000



improvement: biolog. knowledge, molecular dynamics

$\Rightarrow$ 100



- often: low-resolution model = lattice model

# Previous Prediction Approaches for Latttice Models

- sometimes: heuristic approaches
  - chain growth algorithms
  - genetic algorithms
  - ...

  advantages:
  - fast

  disadvantages:
  - only for structure prediction

- mostly: monte-carlo/simulated annealing

  advantages:
  - easy to adapt
  - if ergodic, then known distribution

  disadvantages:
  - for HP-model, optimal solution nearly never found
  - most approaches are **not** ergodic

- also: complete enumeration

  advantages:
  - direct exploration of landscape

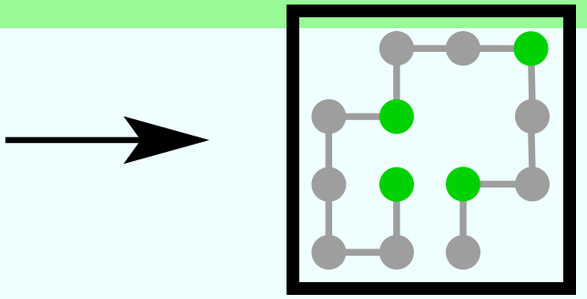
  disadvantages:
  - very short sequences, only 2D

# Monte-Carlo with Simulated Annealing

current = random conf.

# Monte-Carlo with Simulated Annealing

current = random conf.



Initial Conf.

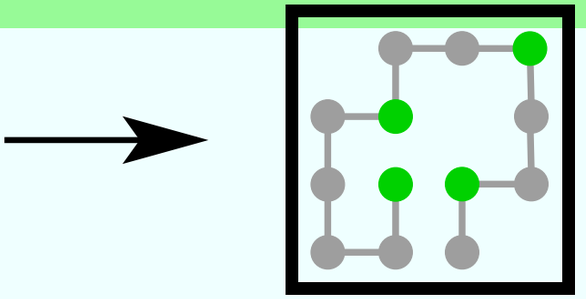
# Monte-Carlo with Simulated Annealing

current = random conf.

while (t < Bound)

new = local-move(current)

$\Delta E$ = #con(new) - #con(current)



Initial Conf.

end

# Monte-Carlo with Simulated Annealing

current = random conf.

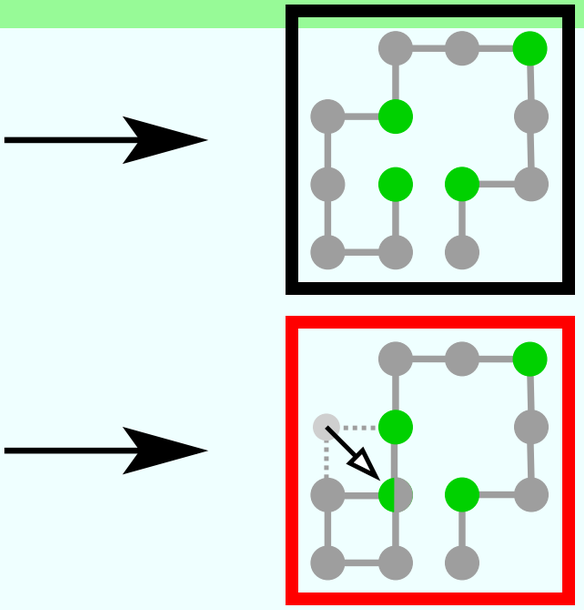
while (t < Bound)

new = local-move(current)

$\Delta E$ = #con(new) - #con(current)

if (not self-avoiding(new)) then

do nothing



Initial Conf.

endif

end

# Monte-Carlo with Simulated Annealing

current = random conf.

while (t < Bound)

new = local-move(current)

$\Delta E$ = #con(new) - #con(current)

if (not self-avoiding(new)) then

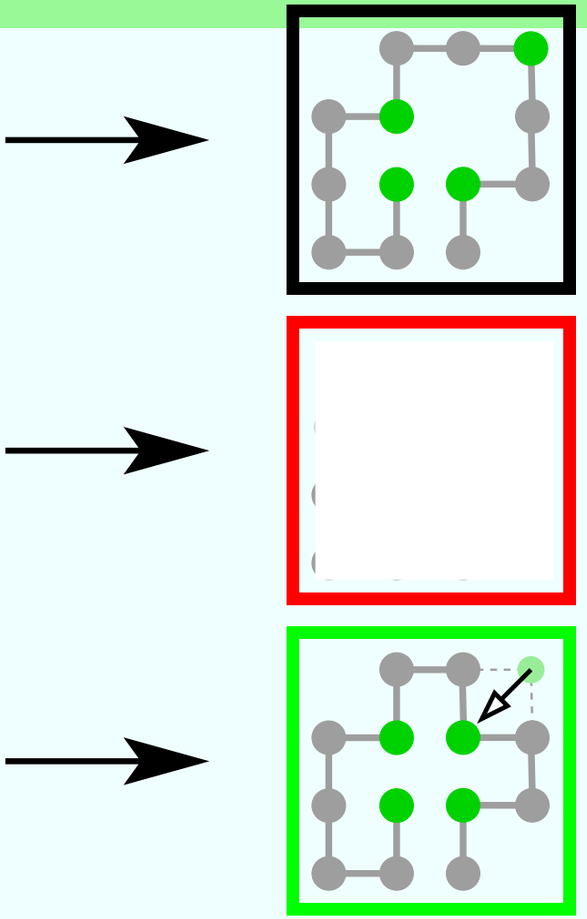
do nothing

elseif ($\Delta E \geq 0$ ) then

current = new

endif

end



Initial Conf.

# Monte-Carlo with Simulated Annealing

current = random conf.

while (t < Bound)

new = local-move(current)

$\Delta E$ = #con(new) - #con(current)

if (not self-avoiding(new)) then

do nothing

elseif ($\Delta E \geq 0$ ) then

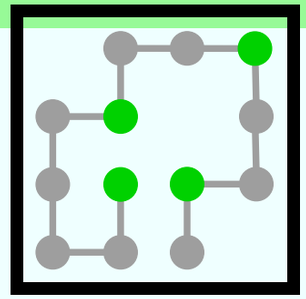
current = new

elseif ($Rand \leq e^{\frac{\Delta E}{k_B T}}$ ) then

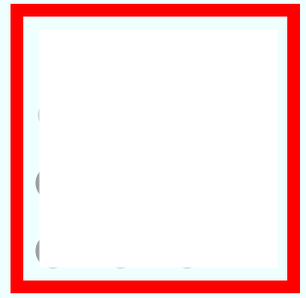
current = new

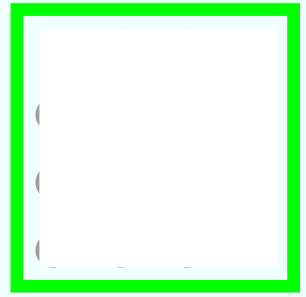
endif

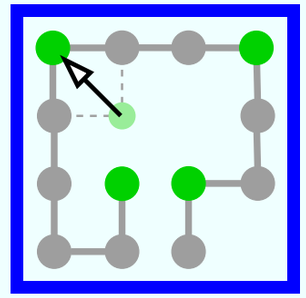
end



Initial Conf.

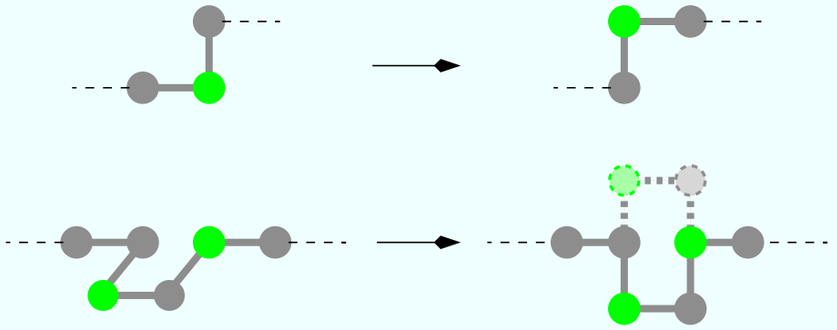






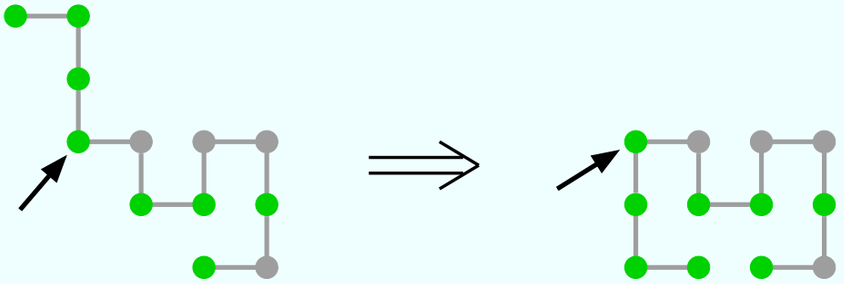
# Move Sets

- often: local moves

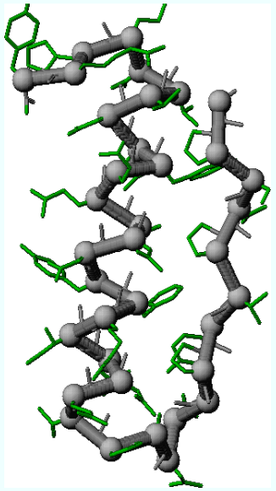


  - fast, but not ergodic
  - what can be said on landscape if not ergodic

- ergodic, but seldomly used (slow)



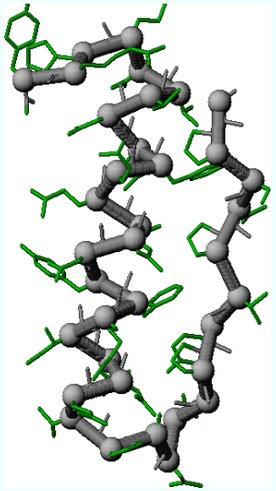
# Idea of Lattice Models



- trade-off: choose between
  - models, that **closely** resembles proteins structure

    **BUT** no hope of ever finding the native structure

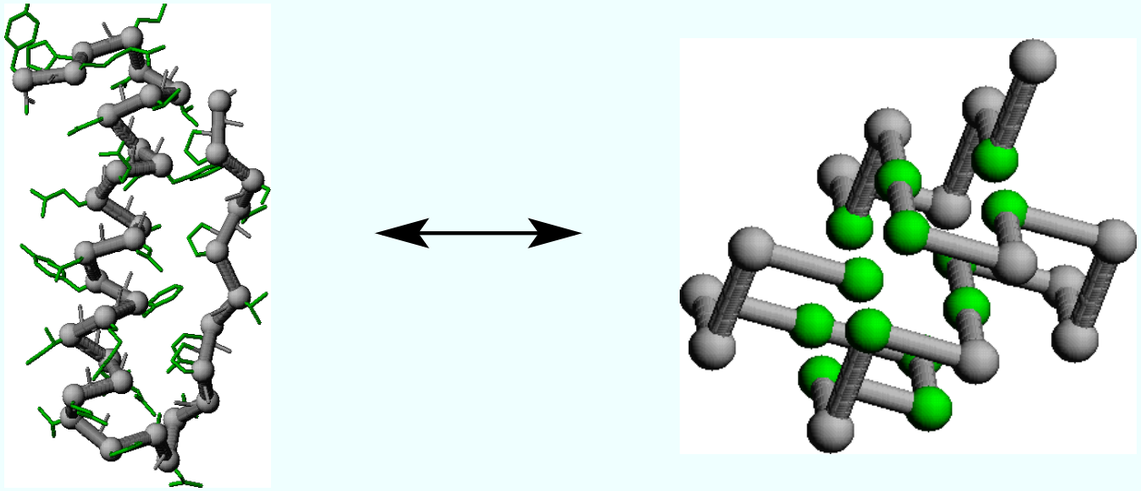
# Idea of Lattice Models



- trade-off: choose between
  - models, that **closely** resembles proteins structure

    **BUT** no hope of (algorithmically) finding the native structure

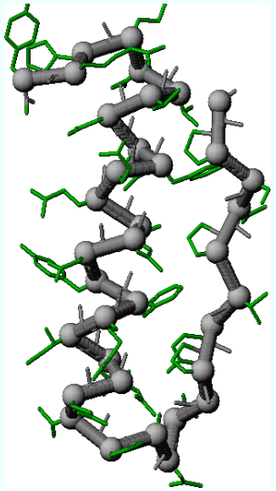
# Idea of Lattice Models

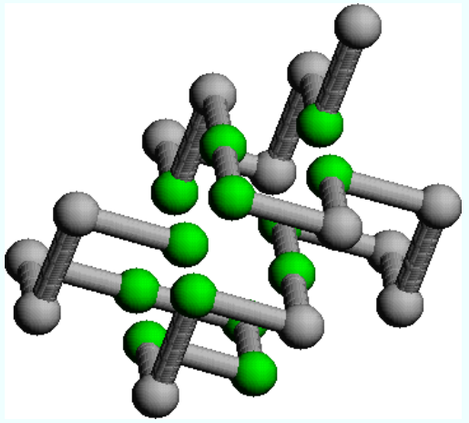


- trade-off: choose between
  - models, that **closely** resembles proteins structure

    **BUT** no hope of (algorithmically) finding the native structure
  - models, that **crudely** resembles proteins structure

# Idea of Lattice Models



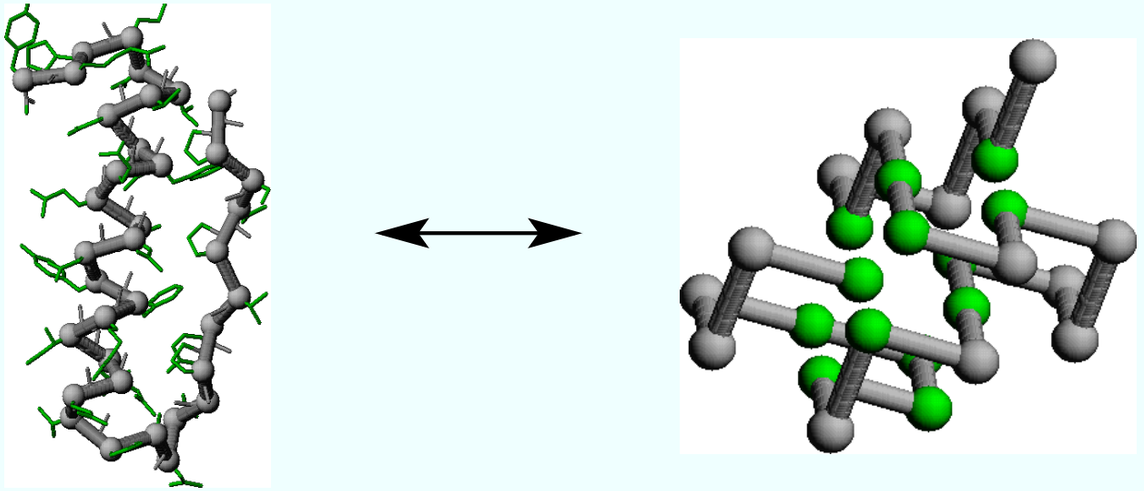


- trade-off: choose between
  - models, that **closely** resembles proteins structure

    **BUT** no hope of (algorithmically) finding the native structure
  - models, that **crudely** resembles proteins structure

    **BUT** we can find the native structure

# Idea of Lattice Models



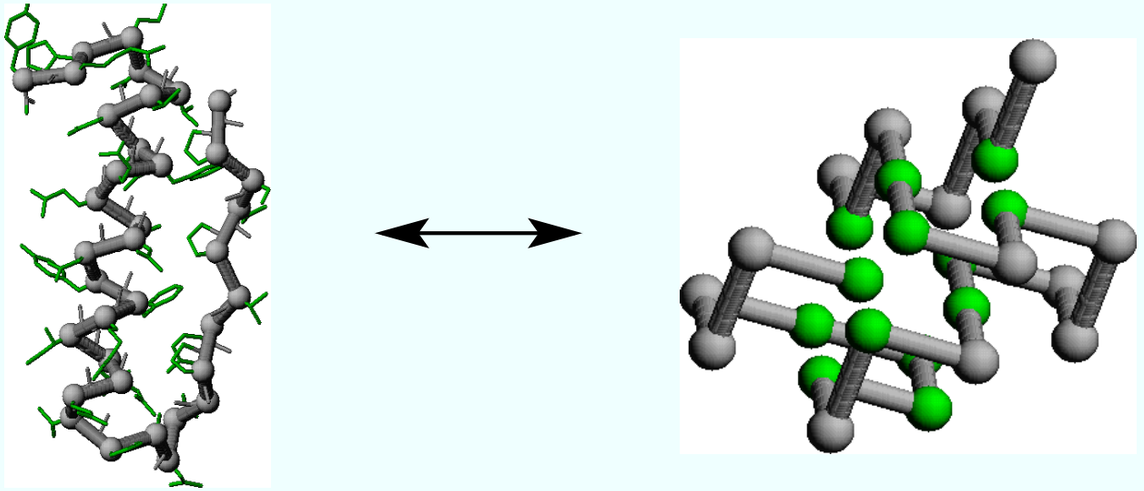
- trade-off: choose between
  - models, that **closely** resembles proteins structure

    **BUT** no hope of (algorithmically) finding the native structure
  - models, that **crudely** resembles proteins structure

    ~~**BUT** we can find the native structure~~

    **BUT SO FAR:** we cannot find the native structure either

# Idea of Lattice Models



- trade-off: choose between
  - models, that **closely** resembles proteins structure

    **BUT** no hope of (algorithmically) finding the native structure
  - models, that **crudely** resembles proteins structure

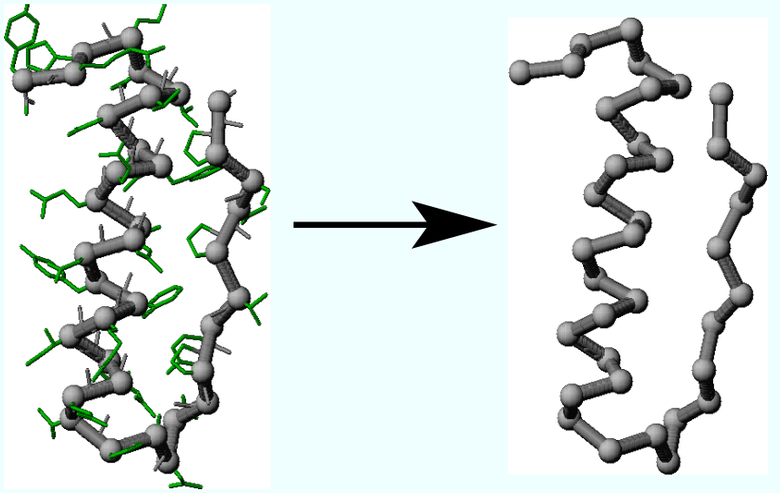
    ~~**BUT** we can find the native structure~~

    **BUT SO FAR:** we cannot find the native structure either
- **here:** **BUT** we can find the native structure

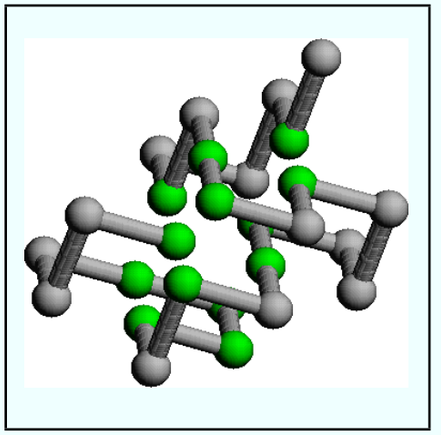
  using constraint programming

# Lattice Models

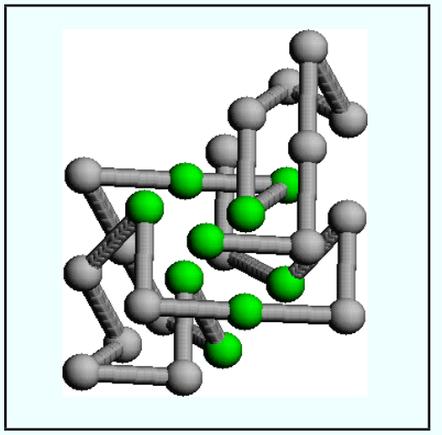
- lattice models:
  - usually only backbone
  - positions = positions on lattice
  - self-avoiding: no steric conflicts



- often used lattices:



cubic

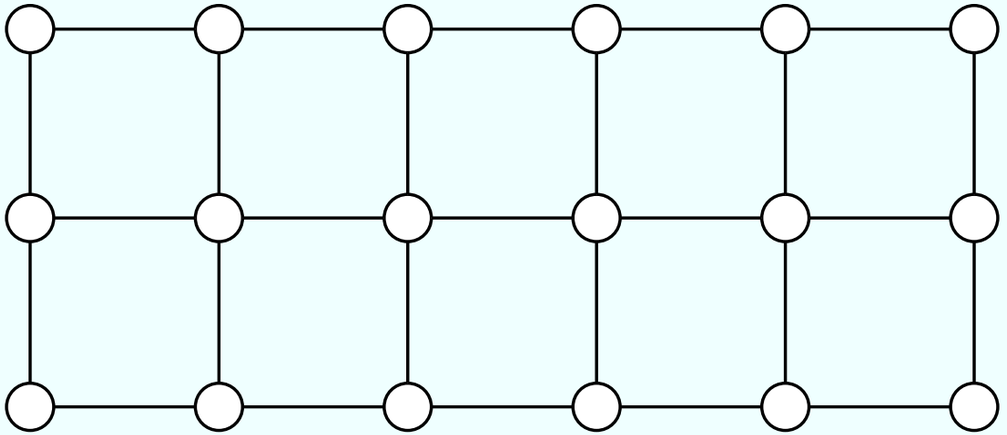


face-centered-cubic

- **BUT:** search for native conformation = NP-complete
- which lattice should be used?

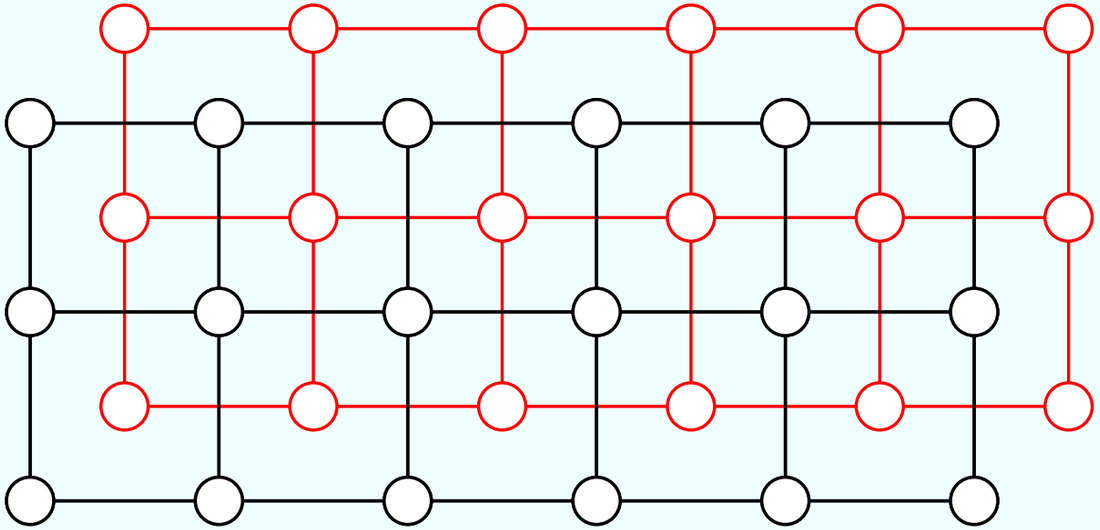
# The FCC

- FCC = face-centered cubic lattice



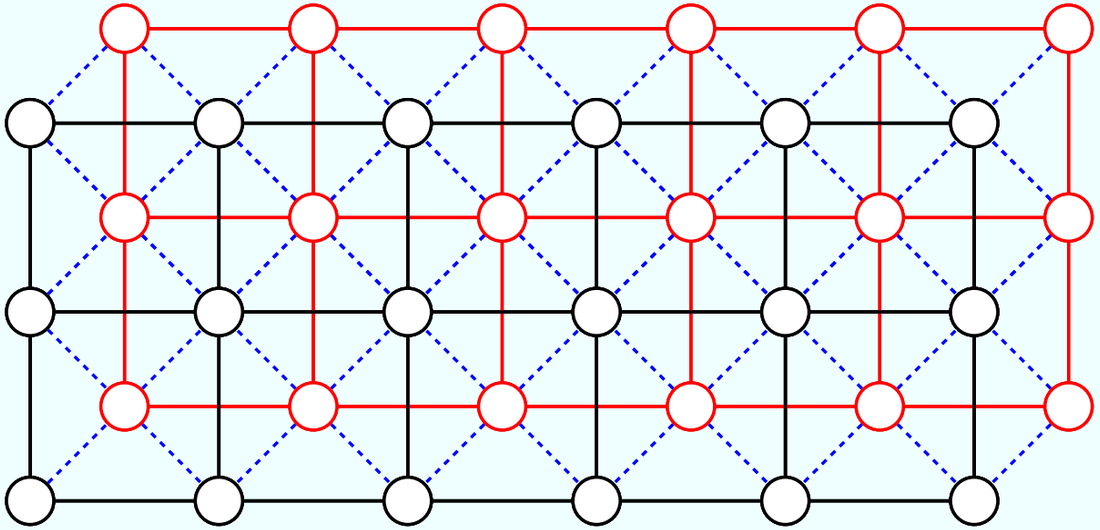
# The FCC

- FCC = face-centered cubic lattice



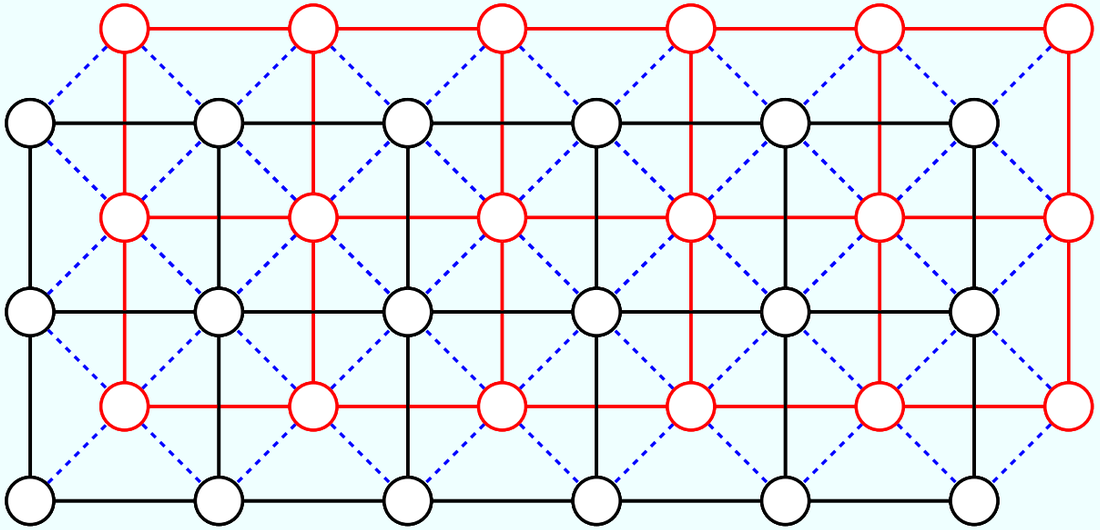
# The FCC

- FCC = face-centered cubic lattice



# The FCC

- FCC = face-centered cubic lattice

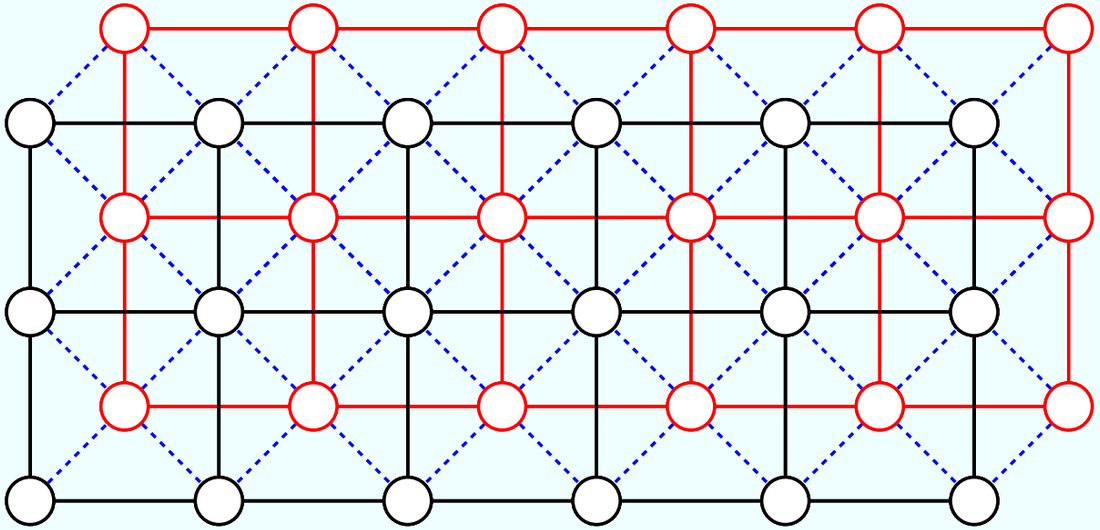


- Kepler's conjecture: FCC=densest packing of balls

  proved just recently (after $\approx 400$ years)

# The FCC

- FCC = face-centered cubic lattice



- Kepler's conjecture: FCC=densest packing of balls

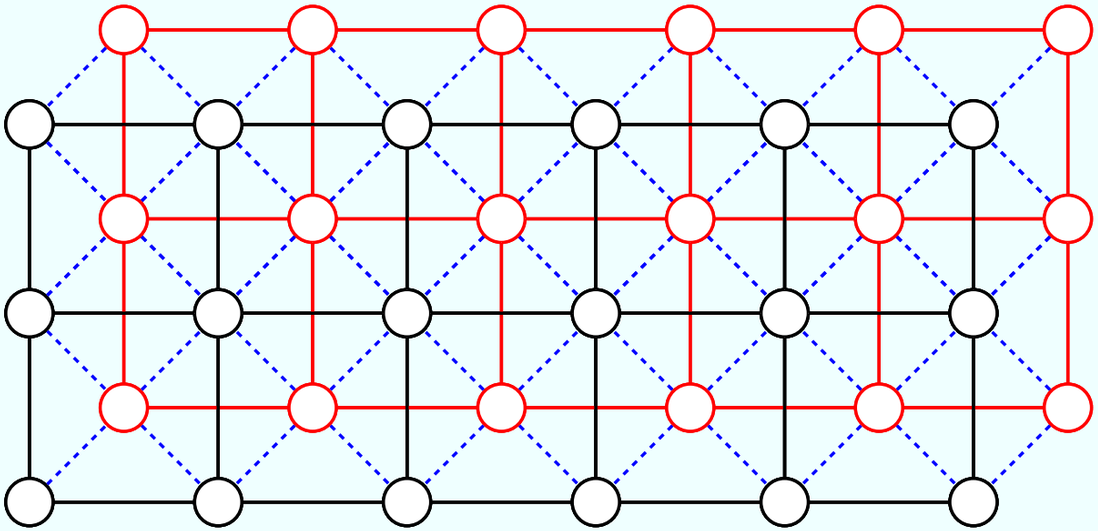
  proved just recently (after $\approx 400$ years)

- [Bagci,Jernigan,Bahar 2002]: clusters of near neighbours in proteins

  *. . . the neighbours are not distributed in a uniform, less dense way, but rather in a clustered dense way, occupying positions that closely approximate those of a distorted FCC packing.. . .*

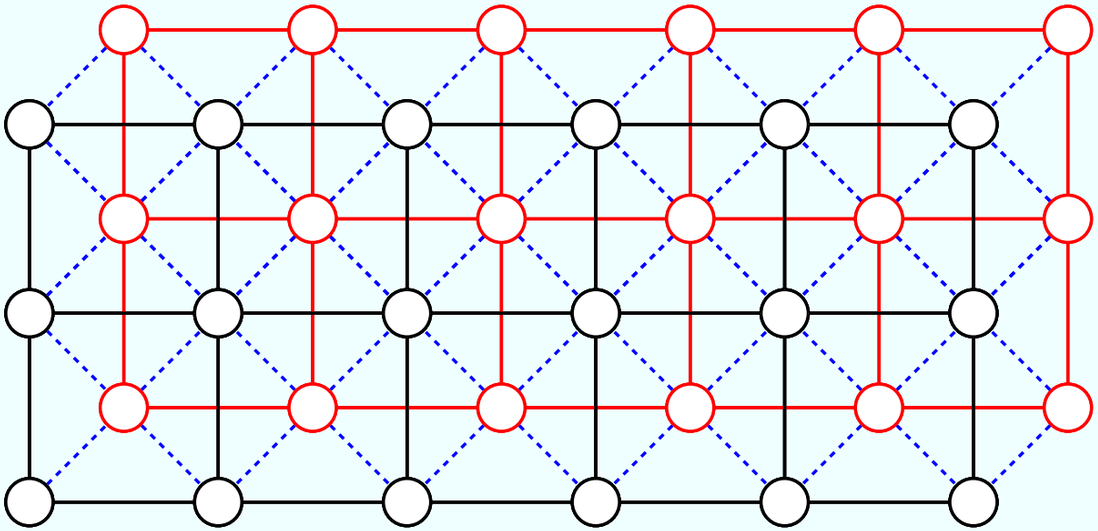
  *. . . We confirm that lattices with large combination numbers provide better fits to protein structures . . .*

# Relation to Proteins



- how does it related to proteins?

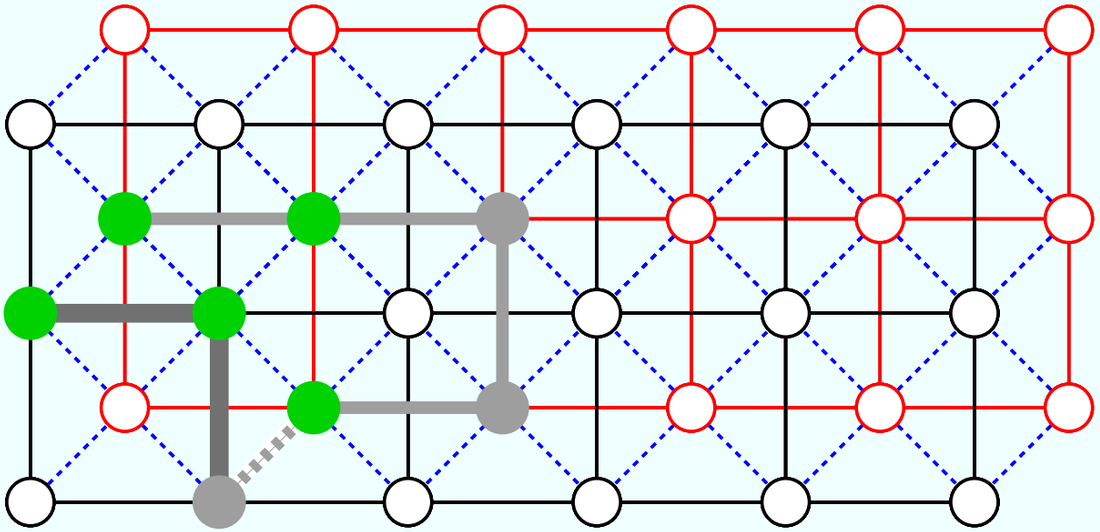
# Relation to Proteins



- how does it related to proteins?
  - hydrophobic and polar (hydrophilic) amino acids
  - hydrophobic are densely packed

alphabet:
**H** = **H**ydrophobic
**P** = **P**olar (hydrophilic)

# Relation to Proteins

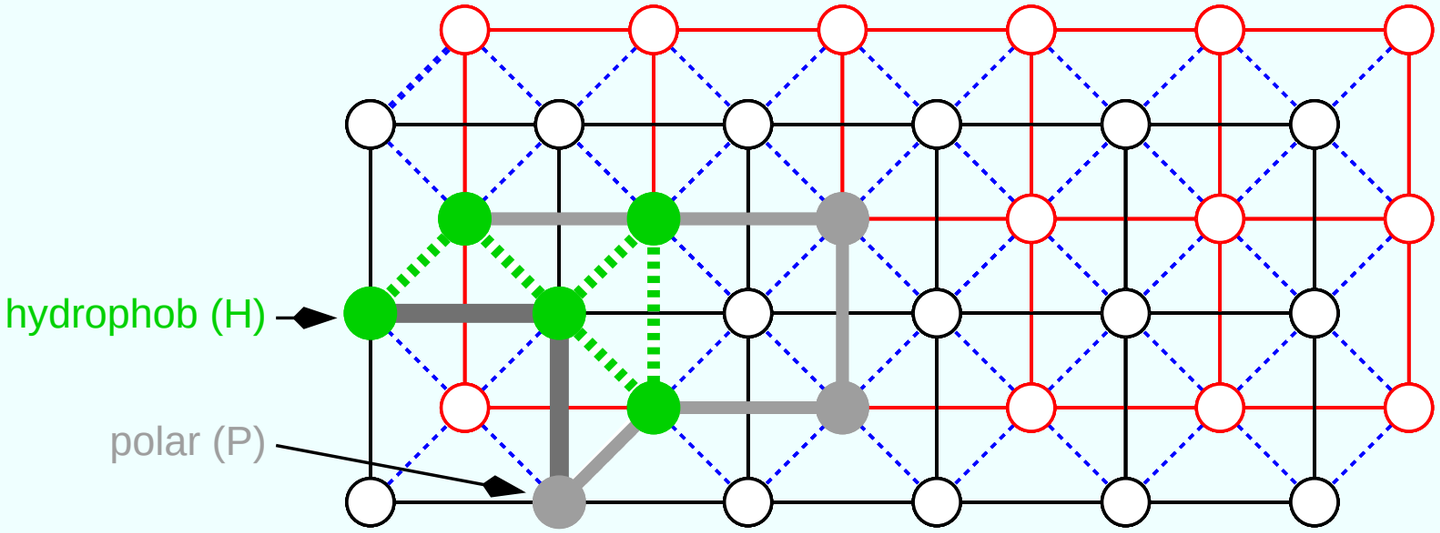


- how does it related to proteins?
  - hydrophobic and polar (hydrophilic) amino acids
  - hydrophobic are densely packed

alphabet:
**H** = **H**ydrophobic
**P** = **P**olar (hydrophilic)

- in the following:
  - search for conformation with densest hydrophobic packing

# Relation to Proteins

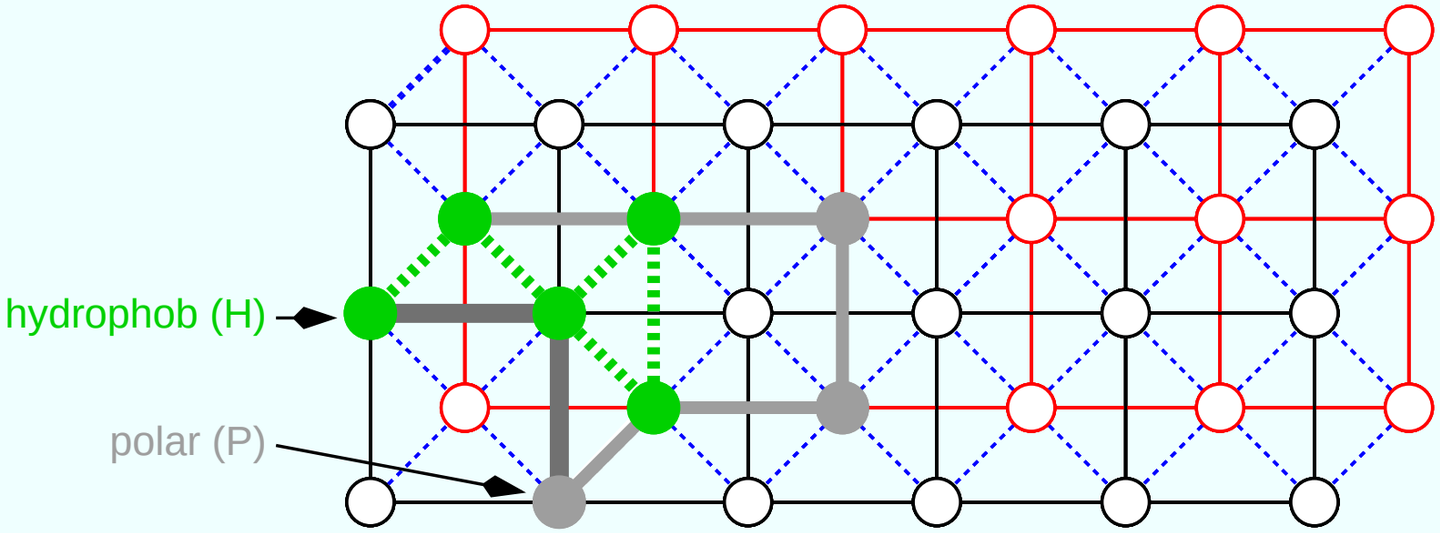



- how does it related to proteins?
  - hydrophobic and polar (hydrophilic) amino acids
  - hydrophobic are densely packed

alphabet:
**H** = **H**ydrophobic
**P** = **P**olar (hydrophilic)

- in the following:
  - search for conformation with densest hydrophobic packing
    = max. number of contacts between hydrophobic AA (green)

# Relation to Proteins




- how does it related to proteins?
  - hydrophobic and polar (hydrophilic) amino acids
  - hydrophobic are densely packed

alphabet:
**H** = **H**ydrophobic
**P** = **P**olar (hydrophilic)

- in the following:
  - search for conformation with densest hydrophobic packing
    = max. number of contacts between hydrophobic AA (green)
  - **HP-model** of Ken Dill: folding of sequences consisting of H and P

# General Approach

- Algorithm consist of three steps:
- Step 1 and 2 are precomputation steps

# General Approach

- Algorithm consist of three steps:
- Step 1 and 2 are precomputation steps

Step 1: compute lower energy bounds

estimate contacts (within layers, between layers)

$\Rightarrow$ Step 1

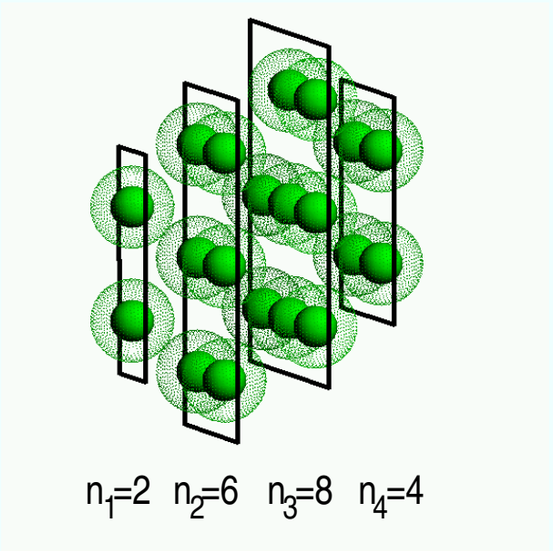
# General Approach

- Algorithm consist of three steps:
- Step 1 and 2 are precomputation steps

Step 1: compute lower energy bounds

estimate contacts (within layers, between layers)

$\Rightarrow$ Step 1



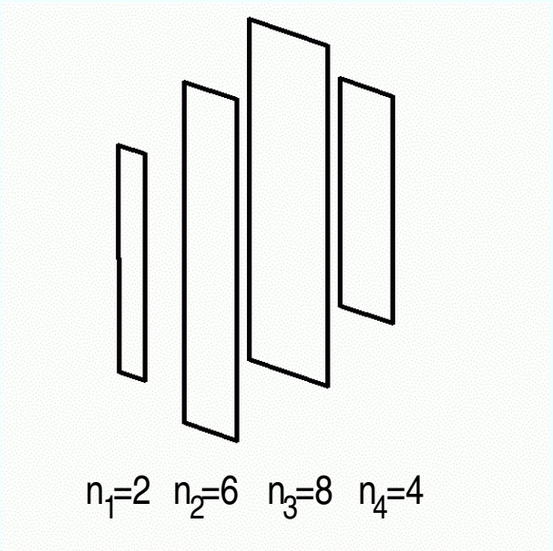
# General Approach

- Algorithm consist of three steps:
- Step 1 and 2 are precomputation steps

Step 1: compute lower energy bounds

estimate contacts (within layers, between layers)

$\Rightarrow$ Step 1



# General Approach

- Algorithm consist of three steps:
- Step 1 and 2 are precomputation steps

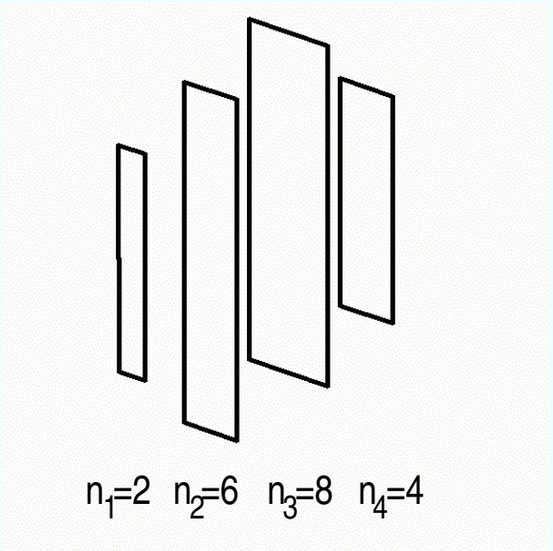
  Step 1: compute lower energy bounds

  estimate contacts (within layers, between layers)

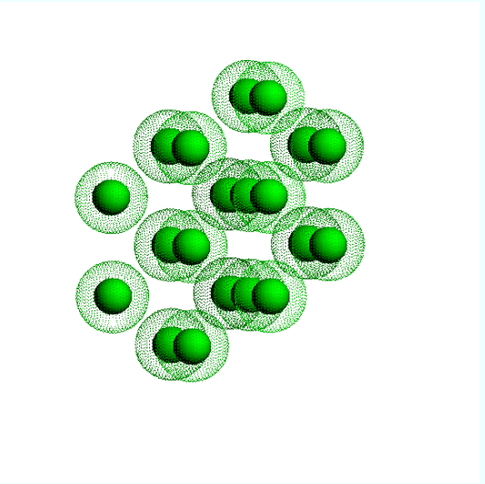
  Step 2: construct hydrophobic cores

  use bounds from last step, precomputed

$\Rightarrow$ Step 1




$\Rightarrow$ Step 2



# General Approach

- Algorithm consist of three steps:
- Step 1 and 2 are precomputation steps

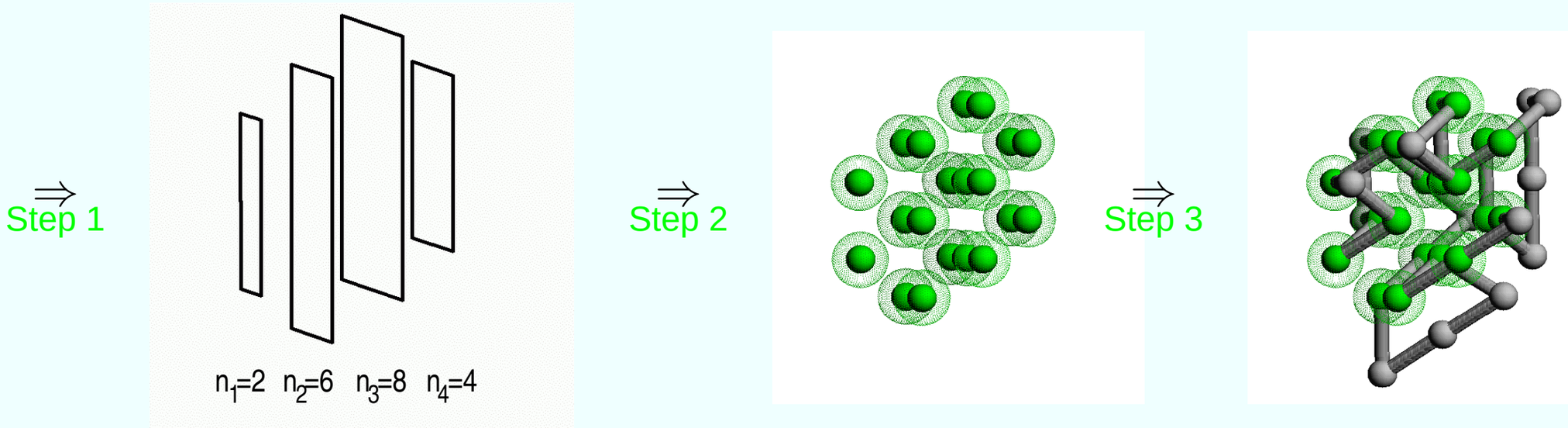
  Step 1: compute lower energy bounds

  estimate contacts (within layers, between layers)

  Step 2: construct hydrophobic cores

  use bounds from last step, precomputed

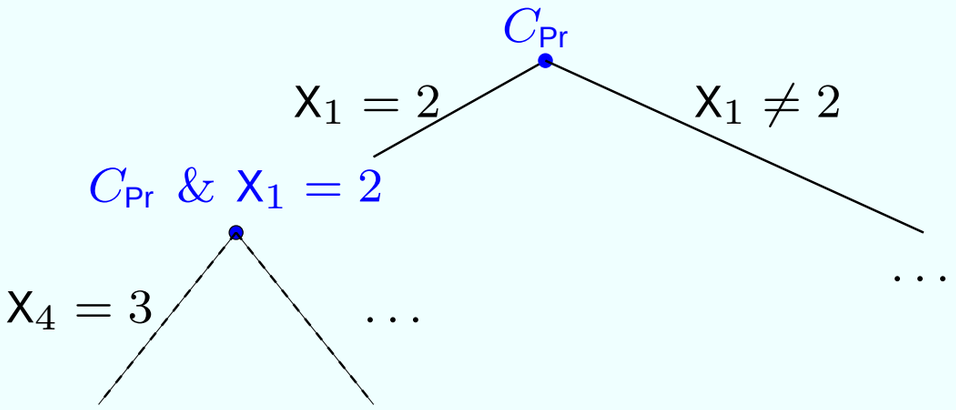
  Step 3: thread sequence to hydrophobic cores of size $n$.



# Constraint-based Formulation

- **constraint problem** $C_{\mathsf{Pr}}$:
  - position of i-th amino acid: $\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i \in [1 \ldots n]$
  - constraints describe Self-Avoiding Walks

    $(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) \neq (\mathsf{X}_j, \mathsf{Y}_j, \mathsf{Z}_j)$ and $|(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) - (\mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})| = 1$
- **constraint-based optimization**: distributing over aminoacid positions

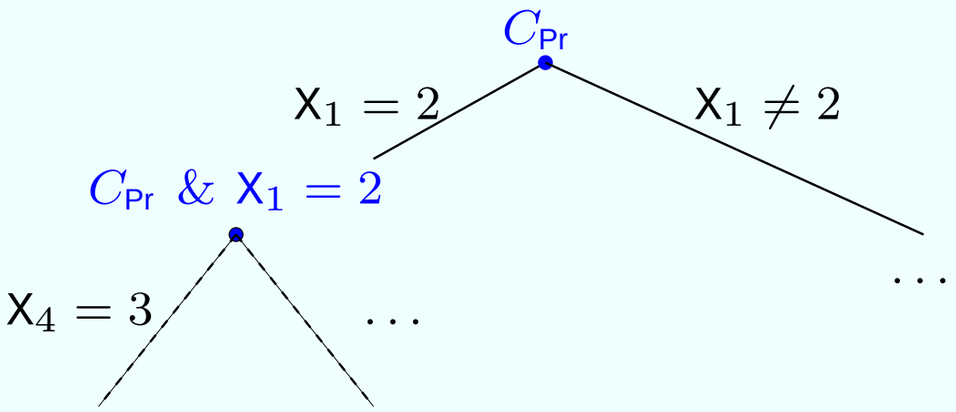


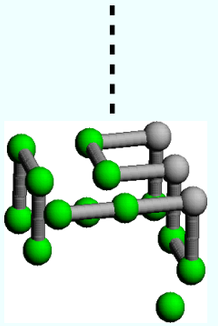
# Constraint-based Formulation

- **constraint problem** $C_{\mathsf{Pr}}$:
  - position of i-th amino acid: $\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i \in [1 \ldots n]$
  - constraints describe Self-Avoiding Walks

$$(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) \neq (\mathsf{X}_j, \mathsf{Y}_j, \mathsf{Z}_j) \quad \text{and} \quad |(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) - (\mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})| = 1$$

- **constraint-based optimization**: distributing over aminoacid positions



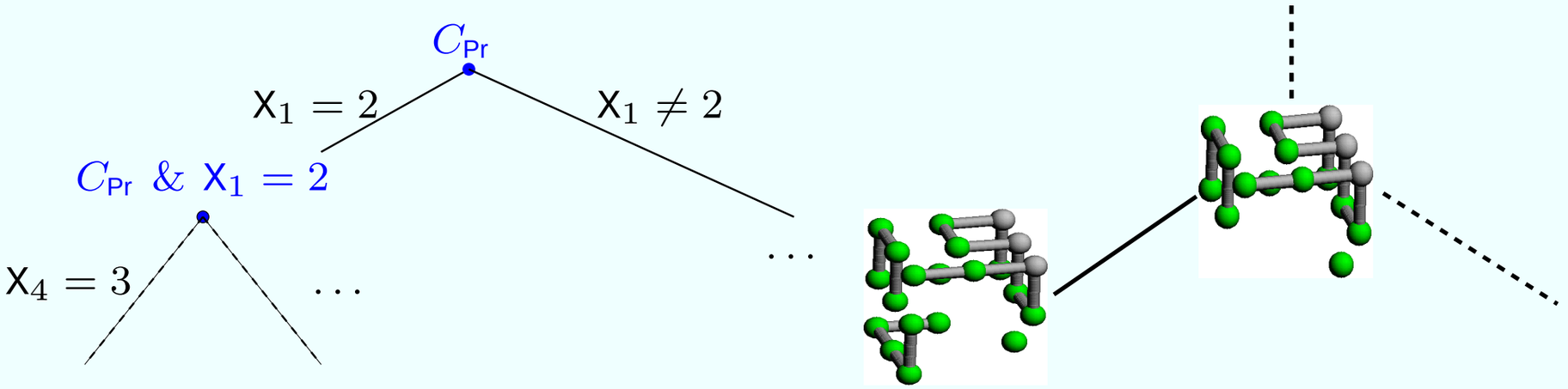



# Constraint-based Formulation

- **constraint problem** $C_{\mathsf{Pr}}$:
  - position of i-th amino acid: $\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i \in [1 \ldots n]$
  - constraints describe Self-Avoiding Walks

$$(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) \neq (\mathsf{X}_j, \mathsf{Y}_j, \mathsf{Z}_j) \quad \text{and} \quad |(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) - (\mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})| = 1$$

- **constraint-based optimization**: distributing over aminoacid positions

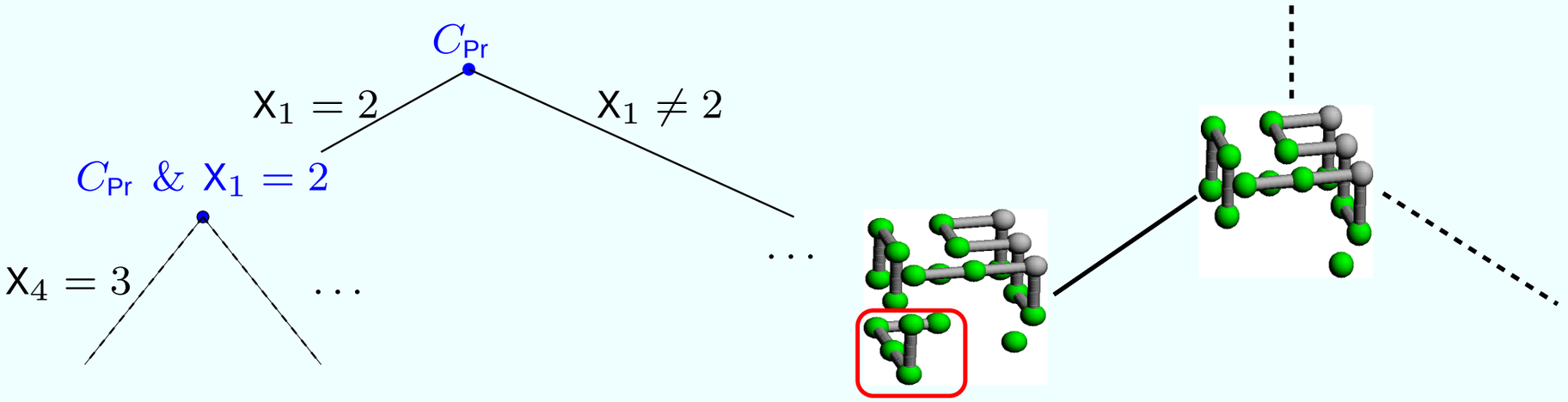


# Constraint-based Formulation

- **constraint problem** $C_{\mathsf{Pr}}$:
  - position of i-th amino acid: $\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i \in [1 \ldots n]$
  - constraints describe Self-Avoiding Walks

$$(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) \neq (\mathsf{X}_j, \mathsf{Y}_j, \mathsf{Z}_j) \quad \text{and} \quad |(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) - (\mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})| = 1$$

- **constraint-based optimization**: distributing over aminoacid positions

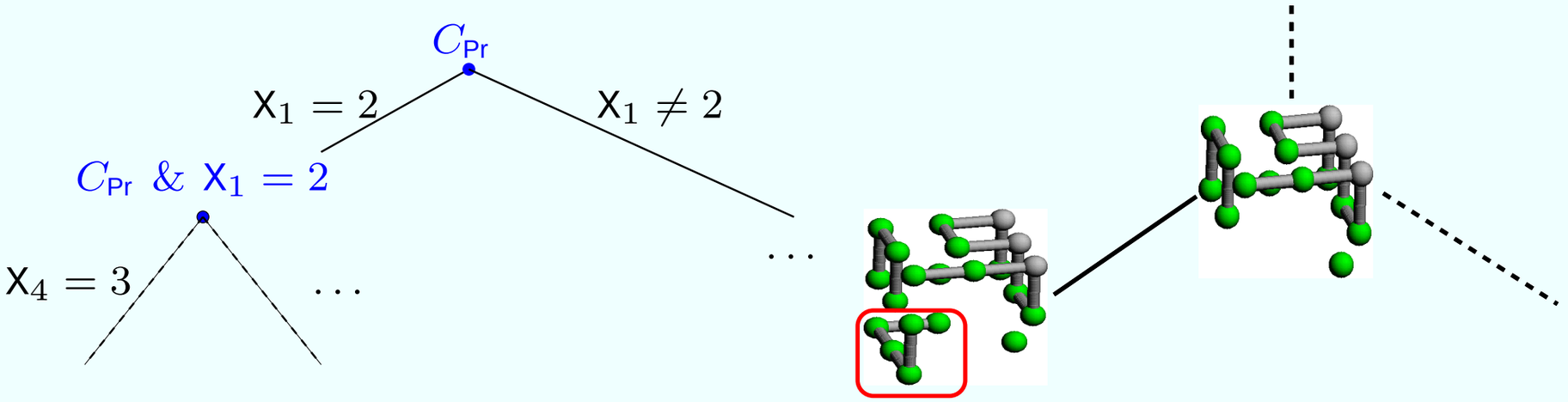


# Constraint-based Formulation

- **constraint problem** $C_{\mathsf{Pr}}$:
  - position of i-th amino acid: $\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i \in [1 \ldots n]$
  - constraints describe Self-Avoiding Walks

$$(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) \neq (\mathsf{X}_j, \mathsf{Y}_j, \mathsf{Z}_j) \quad \text{and} \quad |(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) - (\mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})| = 1$$

- **constraint-based optimization**: distributing over aminoacid positions



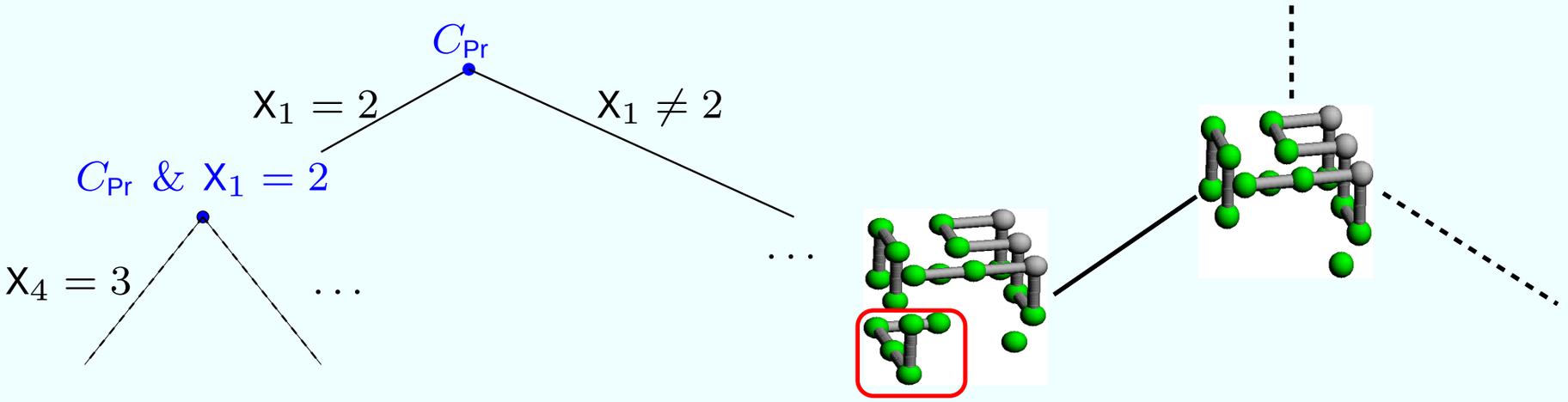

- **problems**
  - redundant constraints and search strategy [Backofen:98]
  - symmetry breaking [Backofen&Will:99]
  - bound for number of HH-contacts [Backofen:00a,03]
  - new constraints, propagation [Backofen:Will:01]

# Constraint-based Formulation

- **constraint problem** $C_{\mathsf{Pr}}$:
  - position of i-th amino acid: $\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i \in [1 \ldots n]$
  - constraints describe Self-Avoiding Walks

$$(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) \neq (\mathsf{X}_j, \mathsf{Y}_j, \mathsf{Z}_j) \quad \text{and} \quad |(\mathsf{X}_i, \mathsf{Y}_i, \mathsf{Z}_i) - (\mathsf{X}_{i+1}, \mathsf{Y}_{i+1}, \mathsf{Z}_{i+1})| = 1$$

- **constraint-based optimization**: distributing over aminoacid positions




- **problems**
  - redundant constraints and search strategy [Backofen:98]
  - symmetry breaking [Backofen&Will:99]
  - **bound for number of HH-contacts [Backofen:00a,03]**
  - new constraints, propagation [Backofen:Will:01]

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  |  | S | E | N | D |
| + |  | M | O | R | E |
| = | M | O | N | E | Y |

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  |  | S | E | N | D |
| + |  | M | O | R | E |
| = | M | O | N | E | Y |

- as a constraint problem:

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- as a constraint problem:

$$S, \ldots, Y \in \{0, \ldots, 9\}$$

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- as a constraint problem:

$$S,\ldots,Y \in \{0,\ldots,9\}$$
$$S \neq 0, \quad M \neq 0$$

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- as a constraint problem:

$$S, \ldots, Y \in \{0, \ldots, 9\}$$

$$S \neq 0, \quad M \neq 0$$

S . . . Y **all different**

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- as a constraint problem:

$$S, \ldots, Y \in \{0, \ldots, 9\}$$

$$S \neq 0, \quad M \neq 0$$

S . . . Y **all different**

$$1000*S + 100*E + 10*N + 1*D$$
$$+ \; 1000*M + 100*O + 10*R + 1*E$$
$$= \; 10000*M + 1000*O + 100*N + 10*E + 1*Y$$

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

| | | | | | |
|---|---|---|---|---|---|
| | | S | E | N | D |
| + | | M | O | R | E |
| = | M | O | N | E | Y |

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:
- **S + M $\geq$ 10*M**

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:
- **S + M $\geq$ 10*M** $\Longrightarrow$ **M = 1**

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:
- **S + M $\geq$ 10*M** $\Longrightarrow$ **M = 1**
- **S + M $\geq$ 10**

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:
- **S + M $\geq$ 10*M** $\Longrightarrow$ **M = 1**
- **S + 1 $\geq$ 10**

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y
  different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:

- **S + M $\geq$ 10*M** $\Longrightarrow$ **M = 1**
- **S + 1 $\geq$ 10** $\Longrightarrow$ **S = 9** ***and*** **O = 0**

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y
  different with

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:

- **S + M $\geq$ 10*M** $\Longrightarrow$ **M = 1**
- **S + 1 $\geq$ 10** $\Longrightarrow$ **S = 9** ***and*** **O = 0**
- from **all different** $\Longrightarrow$ **E, N, D, R, Y** $\in \{2, \ldots, 8\}$

# SEND + MORE = MONEY

- search numbers S, E, N, D, M, O, R, Y different with

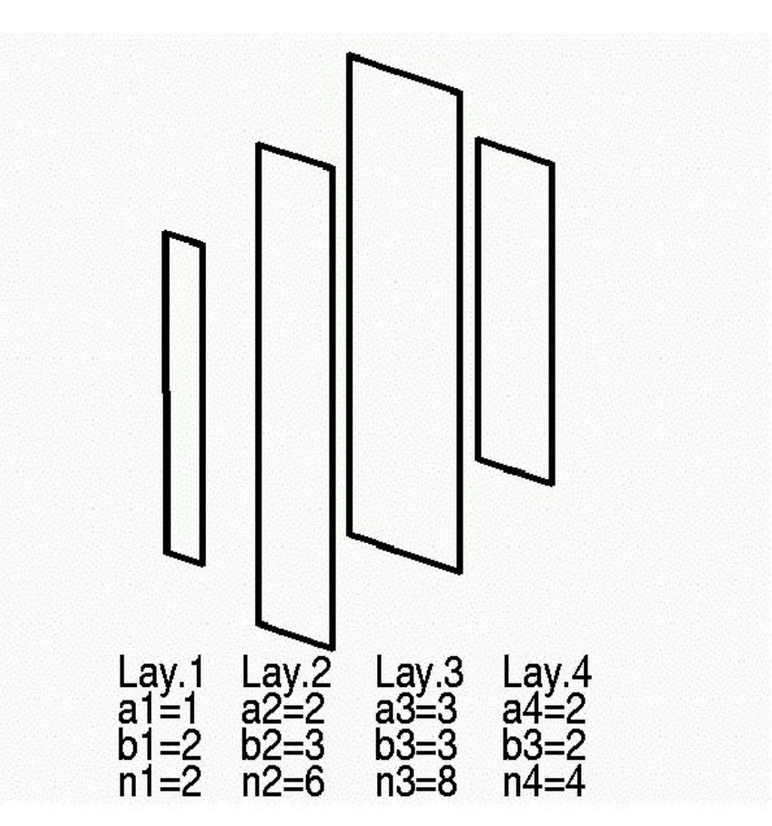
$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline = & M & O & N & E & Y \end{array}$$

- complete enumeration: $\approx 10^8$ combinations $\Longrightarrow$ **Constraint Propagierung**:

- **S + M $\geq$ 10*M** $\Longrightarrow$ **M = 1**
- **S + 1 $\geq$ 10** $\Longrightarrow$ **S = 9** ***and*** **O = 0**
- from **all different** $\Longrightarrow$ **E, N, D, R, Y** $\in \{2, \ldots, 8\}$

**search only for the rest!!**

# Problem 1: Frame Sequences



# Bounds for FCC

- FCC models proteins better: $\sim 1.5 - 2$Å RMSD [Park&Levitt95]

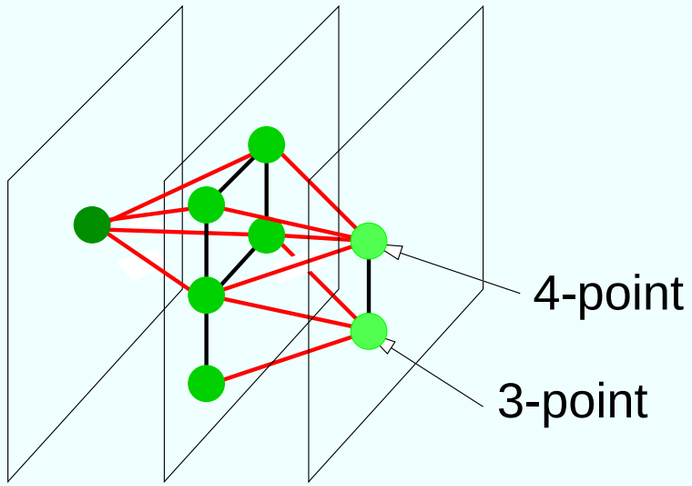
- **BUT**: almost nothing was known
  - approximation: 60% of optimum [Agarwala et al.98]
  - only trivial bounds: 6 $\times$ number of H-amino acids.

# Bounds for FCC

- FCC models proteins better: $\sim 1.5 - 2$Å RMSD [Park&Levitt95]
- **BUT**: almost nothing was known
  - approximation: 60% of optimum [Agarwala et al.98]
  - only trivial bounds: 6 $\times$ number of H-amino acids.
- approach:

layer contacts

interlayer contacts




$x{=}1 \quad x{=}2 \quad x{=}3$

$n_1{=}1 \quad n_2{=}5 \quad n_3{=}2$

# Bounds for FCC

- FCC models proteins better: $\sim 1.5 - 2$Å RMSD [Park&Levitt95]

- **BUT**: almost nothing was known
  - approximation: 60% of optimum [Agarwala et al.98]
  - only trivial bounds: 6 $\times$ number of H-amino acids.

- approach:

  layer contacts

  interlayer contacts

$$\begin{array}{ccc} x{=}1 & x{=}2 & x{=}3 \\ n_1{=}1 & n_2{=}5 & n_3{=}2 \end{array}$$

# Bound on Layer Contact

- relation between surface and contacts

| | | | | |
|---|---|---|---|---|
| 4n | = | 2·H-contacts | + | H-surface |
| ↑ | | | | ↑ |
| number of H-neighbours | | | | contacts to Ps or solution positions |

- relation to frame

# Bound on Layer Contact

- relation between surface and contacts

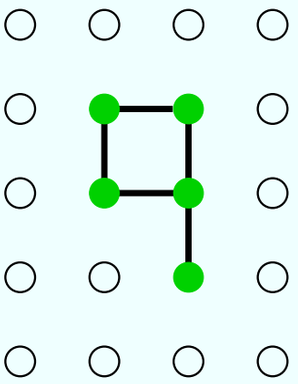
$$4n = 2 \cdot \text{H-contacts} + \text{H-surface}$$

$4n$: number of H-neighbours

H-surface: contacts to Ps or solution positions

- relation to frame

horizontal surface



# Bound on Layer Contact

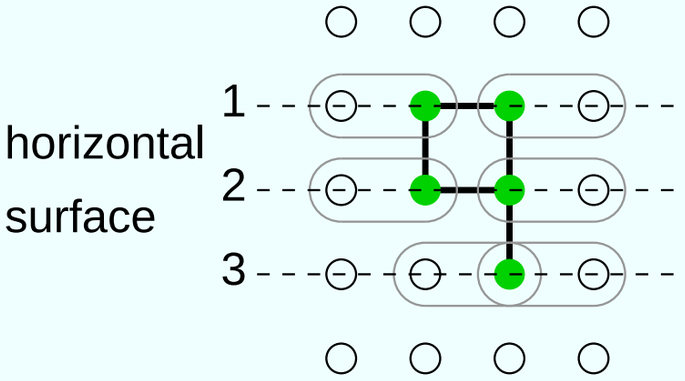
- relation between surface and contacts

$$4n = 2 \cdot \text{H-contacts} + \text{H-surface}$$

$4n$: number of H-neighbours

H-surface: contacts to Ps or solution positions

- relation to frame



# Bound on Layer Contact

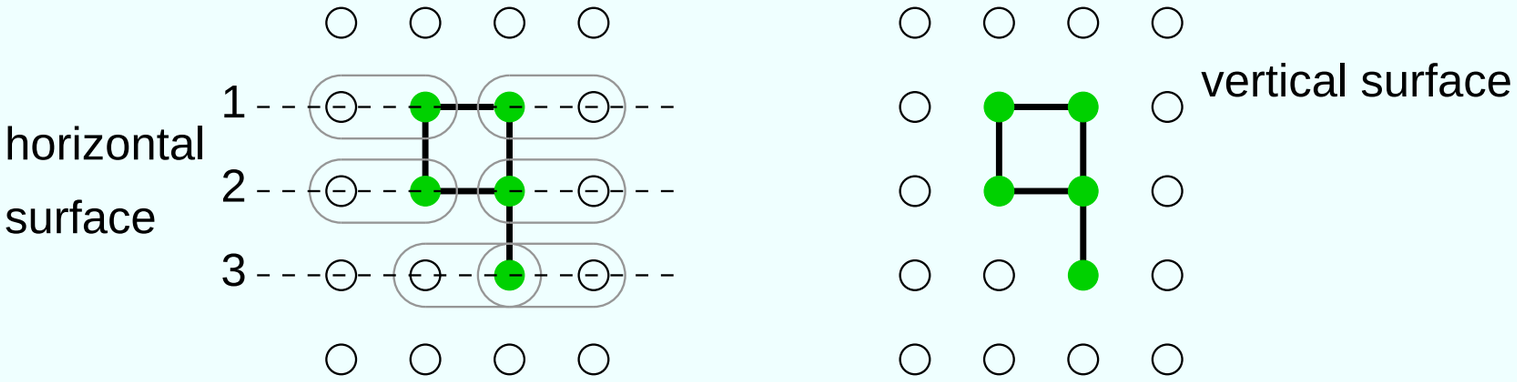
- relation between surface and contacts

$$4n = 2 \cdot \text{H-contacts} + \text{H-surface}$$

$4n$: number of H-neighbours

H-surface: contacts to Ps or solution positions

- relation to frame



# Bound on Layer Contact

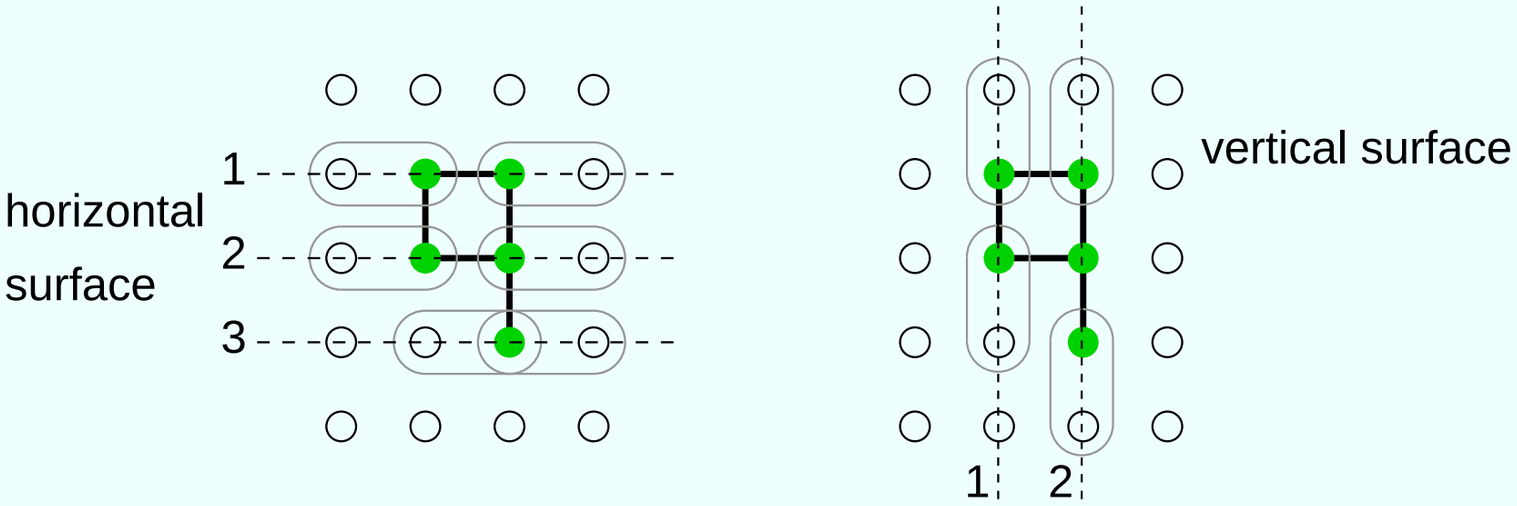
- relation between surface and contacts

$$4n = 2 \cdot \text{H-contacts} + \text{H-surface}$$

$4n$: number of H-neighbours

H-surface: contacts to Ps or solution positions

- relation to frame

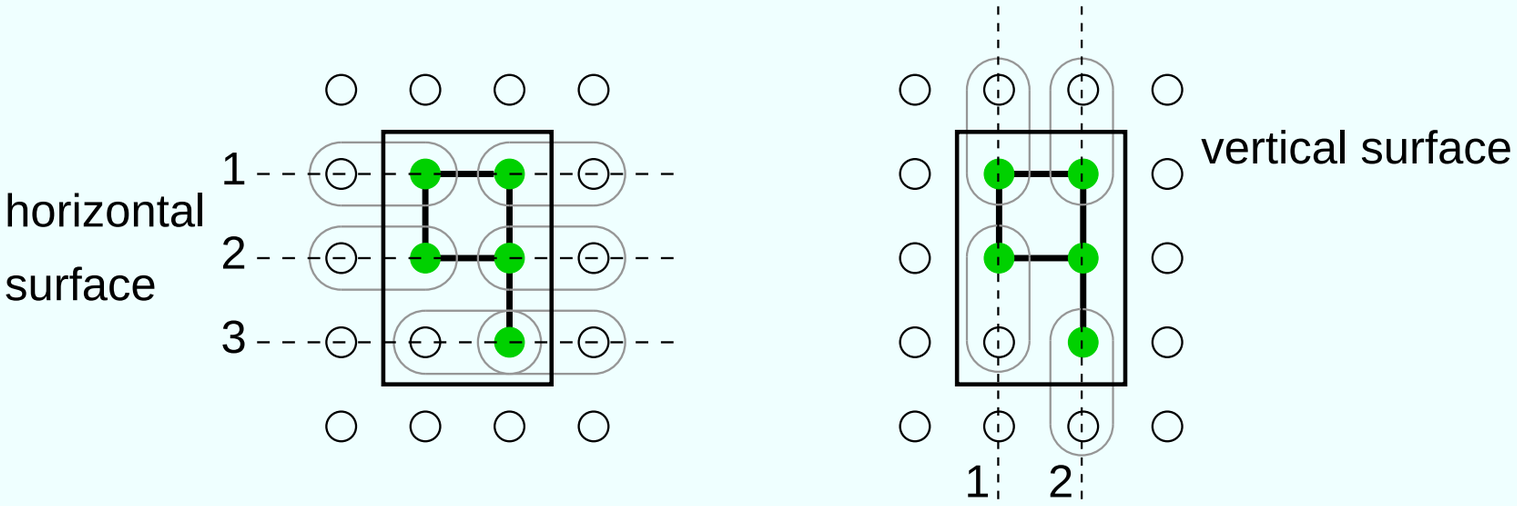


# Bound on Layer Contact

- relation between surface and contacts

| 4n | = | 2·H-contacts | + | H-surface |
|---|---|---|---|---|
| ↑ | | | | ↑ |
| number of H-neighbours | | | | contacts to Ps or solution positions |

- relation to frame H-surface $= 2 \cdot a + 2 \cdot b$ $(a, b)$= (height,width)

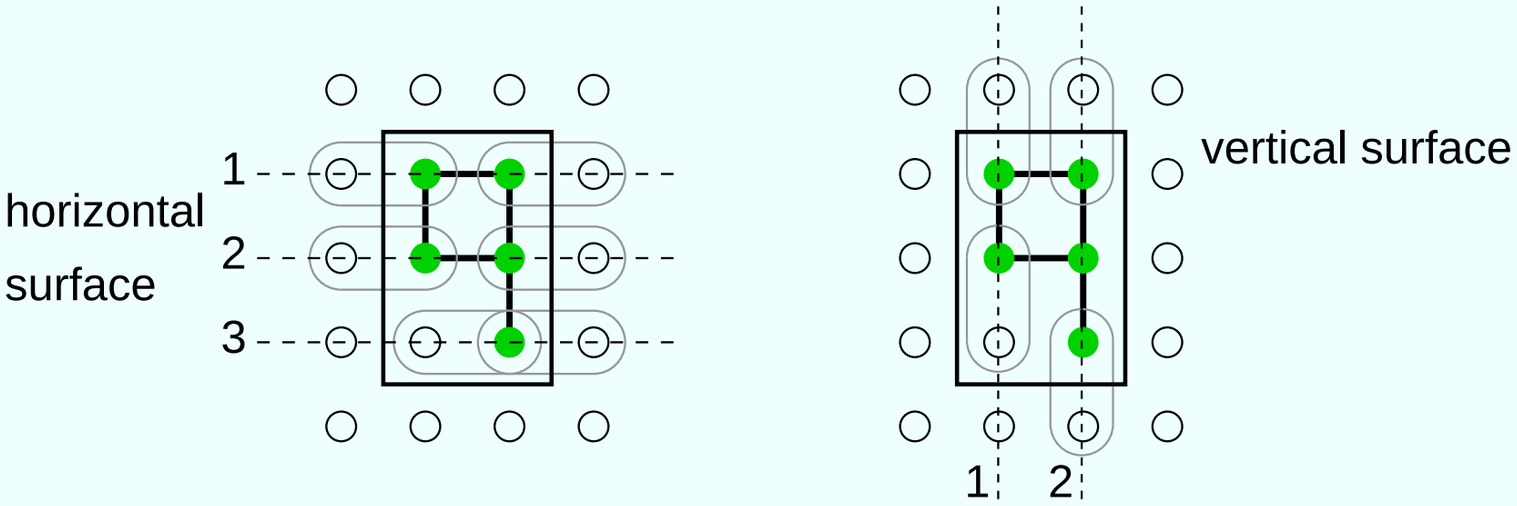


# Bound on Layer Contact

- relation between surface and contacts

| 4n | = | 2·H-contacts | + | H-surface |
|---|---|---|---|---|
| ↑ | | | | ↑ |
| number of H-neighbours | | | | contacts to Ps or solution positions |

- relation to frame H-surface $= 2 \cdot a + 2 \cdot b$ $(a, b)$= (height,width)



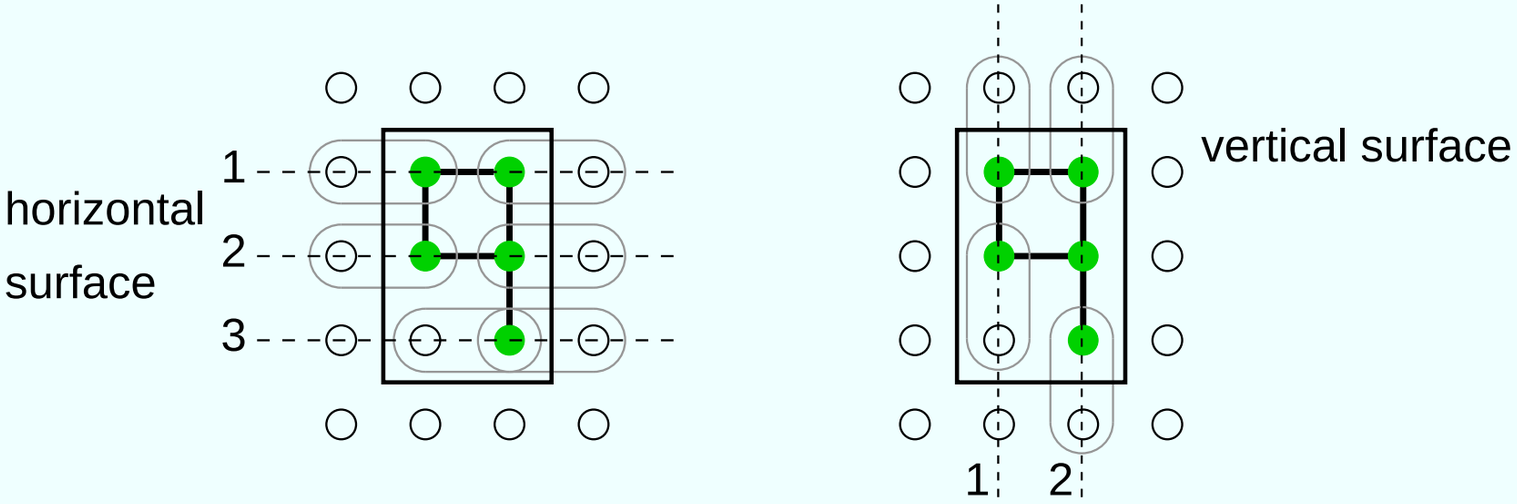

- minimal surface for n Hs $=$ minimal frame $(a, b)$ around n point

# Bound on Layer Contact

- relation between surface and contacts

| 4n | = | 2·H-contacts | + | H-surface |
|---|---|---|---|---|
| ↑ | | | | ↑ |
| number of H-neighbours | | | | contacts to Ps or solution positions |

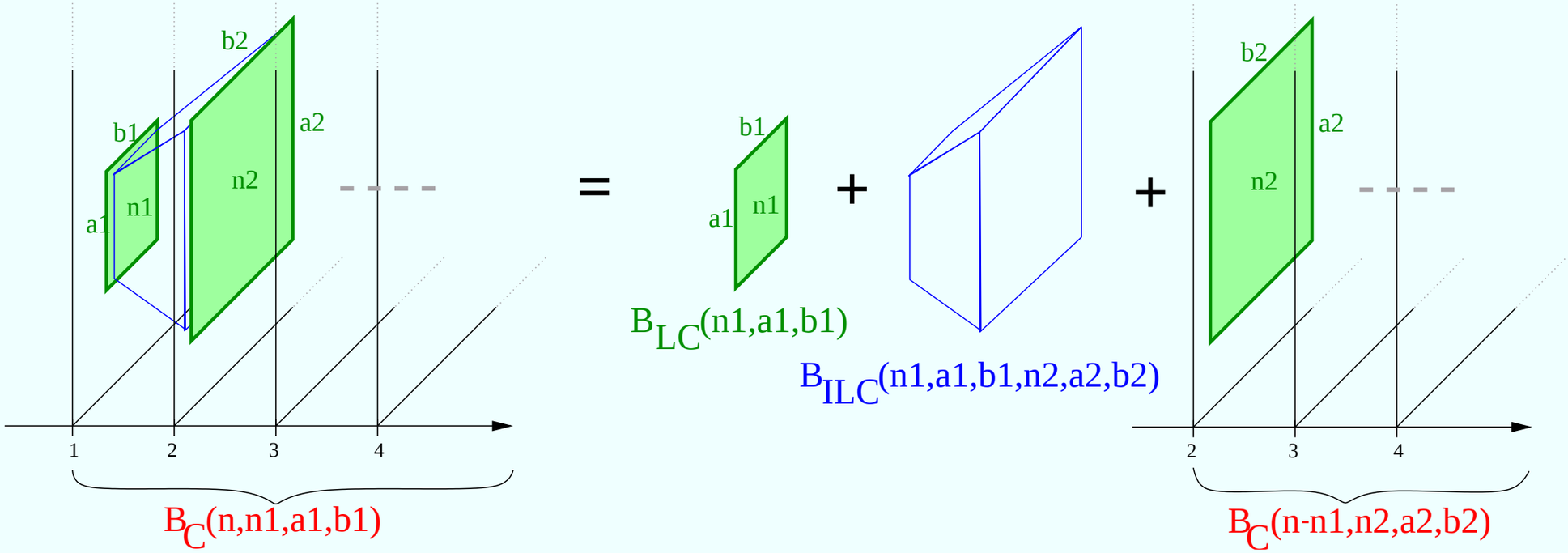
- relation to frame H-surface $= 2 \cdot a + 2 \cdot b$ $(a, b)$= (height,width)




- minimal surface for n Hs = minimal frame $(a, b)$ around n point

$$a = \lceil \sqrt{n} \rceil \qquad b = \lceil \frac{n}{a} \rceil$$

# Recursion for Bound

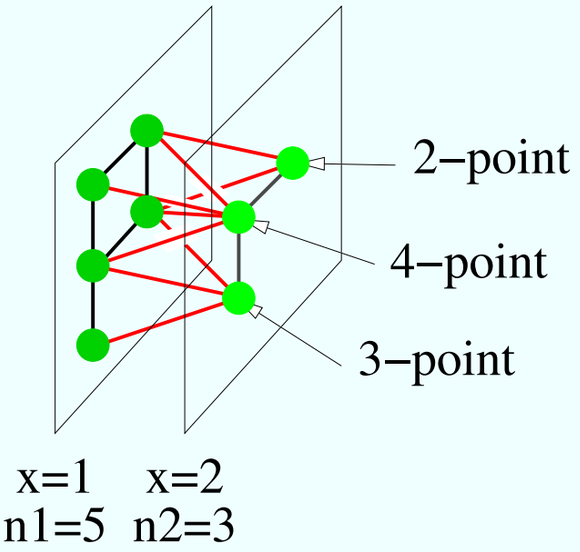



$\mathrm{B_C}(n, n_1, a_1, b_1)$: contacts in core with $n$ elements and first layer $E_1 : n_1, a_1, b_1$

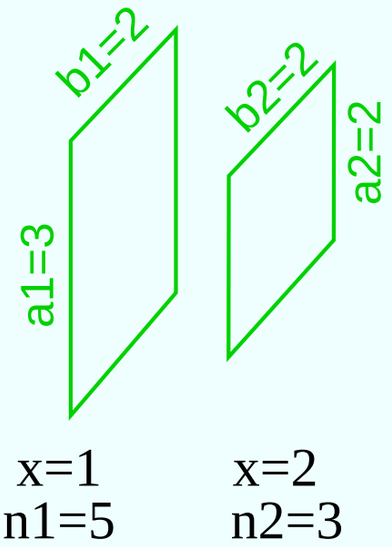
| | | |
|---|---|---|
| = | $\mathrm{B_{LC}}(n_1, a_1, b_1)$ | contacts in layer $E_1$ |
| + | $\mathrm{B_{ILC}}(n_1, a_1, b_1, n_2, a_2, b_2)$ | contacts between layers $E_1$ and $E_2 : n_2, a_2, b_2$ |
| + | $\mathrm{B_C}(n - n_1, n_2, a_2, b_2)$ | contacts in core with $n - n_1$ elements and first layer $E_2$ |

# Bound on Interlayer Contacts

- **recall**: we need an bound on **interlayer contacts**




- **but:** we are given only frames




$\Rightarrow$ bound number of $4-$, $3-$, $2-$ and $1-$points, given frames

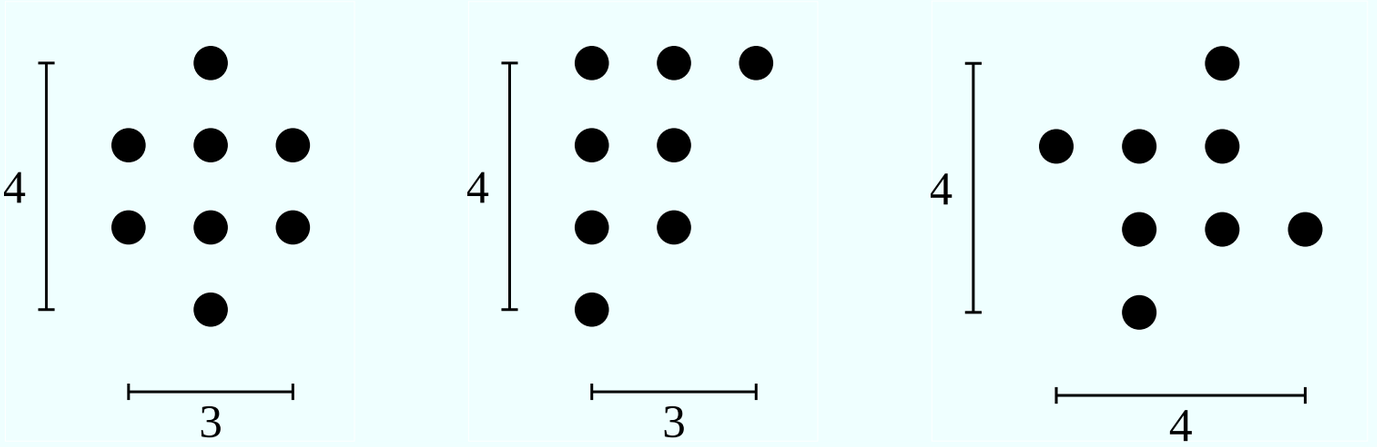
# Bound on Number of 4-, 3-, 2- and 1-Points

- problem: number of 4-, 3-, 2- and 1-points in $x = i + 1$ depends on exact position of Hs in $x = i$

$n_i = 8$

$a_i =$ height

$b_i =$ width




- needed: parameters, which determine the number of 4-, 3-, 2- and 1-points

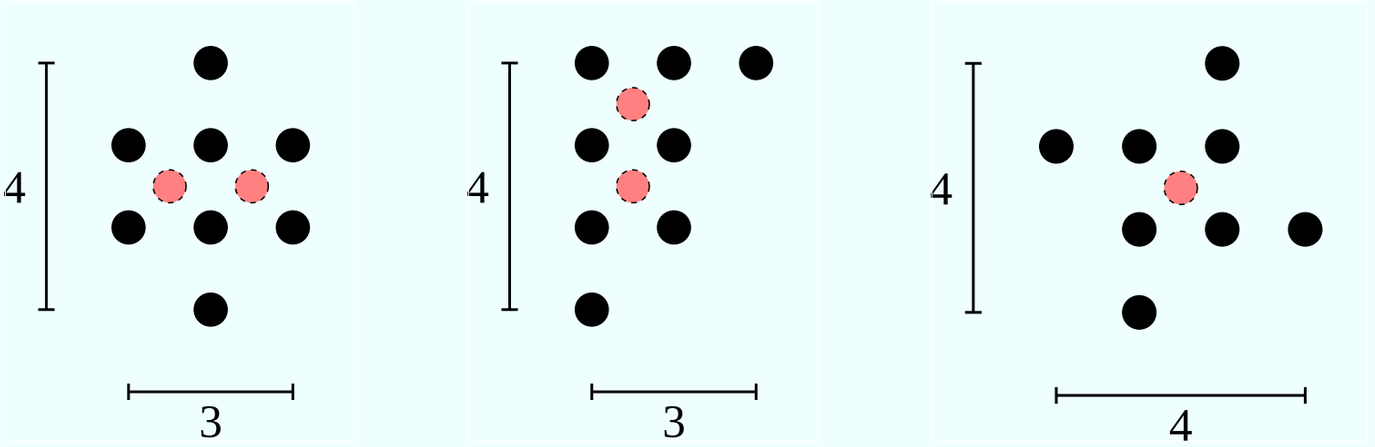
# Bound on Number of 4-, 3-, 2- and 1-Points

- problem: number of 4-, 3-, 2- and 1-points in $x = i + 1$ depends on exact position of Hs in $x = i$

$n_i = 8$

$a_i =$ height

$b_i =$ width




- needed: parameters, which determine the number of 4-, 3-, 2- and 1-points

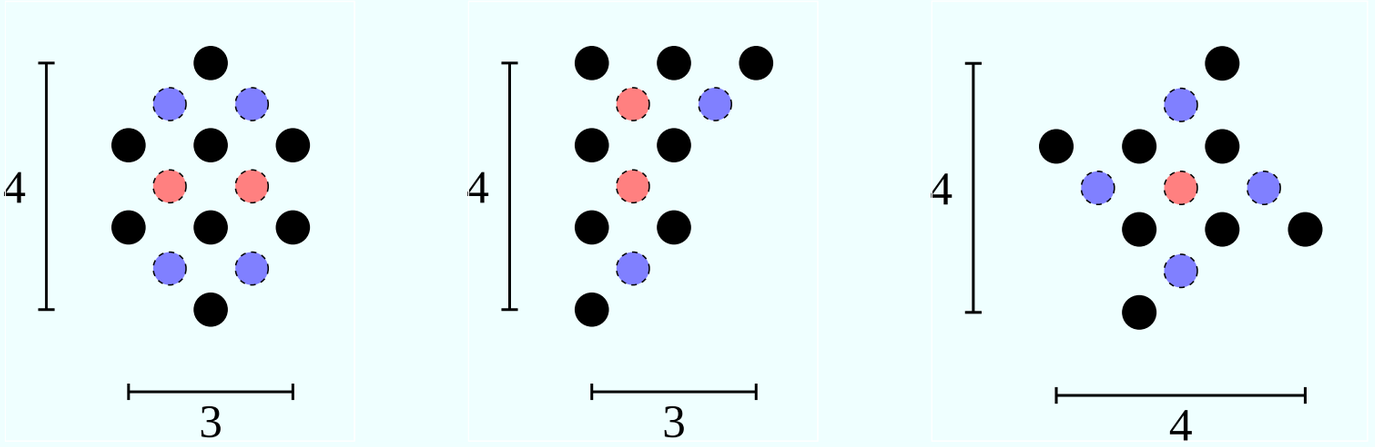
# Bound on Number of 4-, 3-, 2- and 1-Points

- problem: number of 4-, 3-, 2- and 1-points in $x = i + 1$ depends on exact position of Hs in $x = i$

$n_i = 8$

$a_i =$ height

$b_i =$ width




- needed: parameters, which determine the number of 4-, 3-, 2- and 1-points

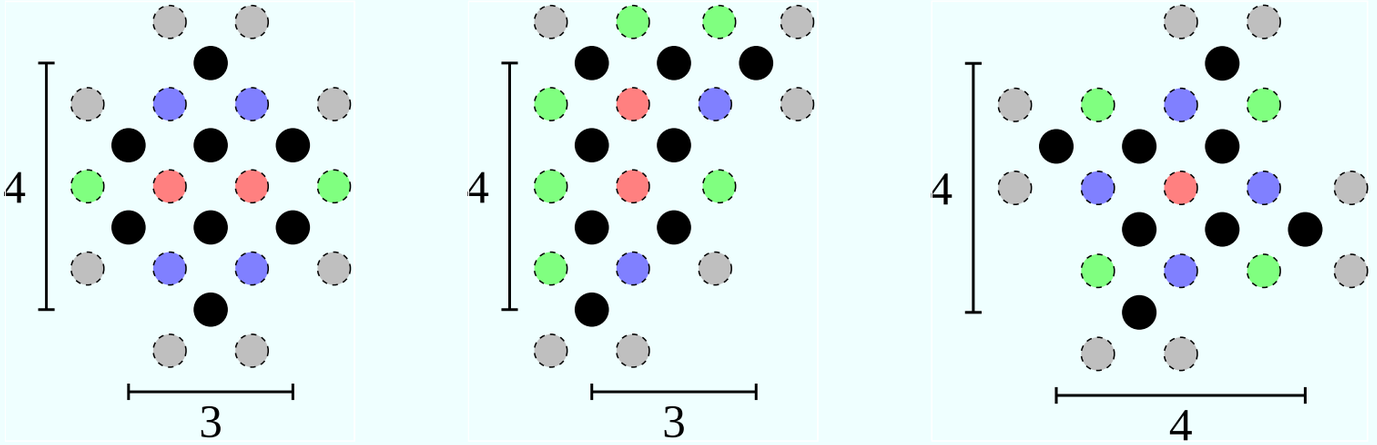
# Bound on Number of 4-, 3-, 2- and 1-Points

- problem: number of 4-, 3-, 2- and 1-points in $x = i + 1$ depends on exact position of Hs in $x = i$

$n_i = 8$

$a_i =$ height

$b_i =$ width




- needed: parameters, which determine the number of 4-, 3-, 2- and 1-points

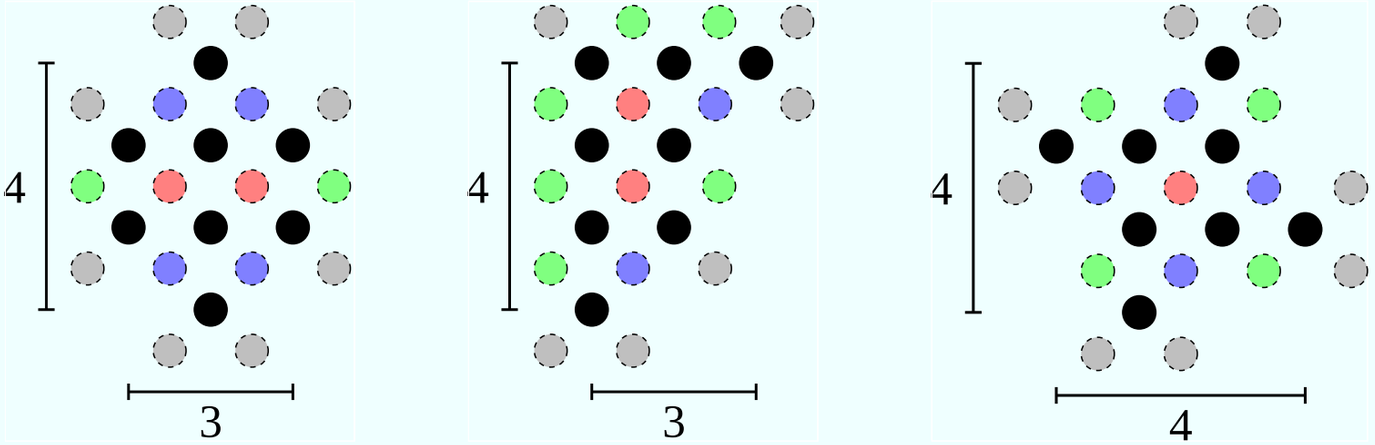
# Bound on Number of 4-, 3-, 2- and 1-Points

- problem: number of 4-, 3-, 2- and 1-points in $x = i + 1$ depends on exact position of Hs in $x = i$

$n_i = 8$

$a_i =$ height

$b_i =$ width




- needed: parameters, which determine the number of 4-, 3-, 2- and 1-points
- **Lemma** let $\ell$ be the number of 3-points. Then:

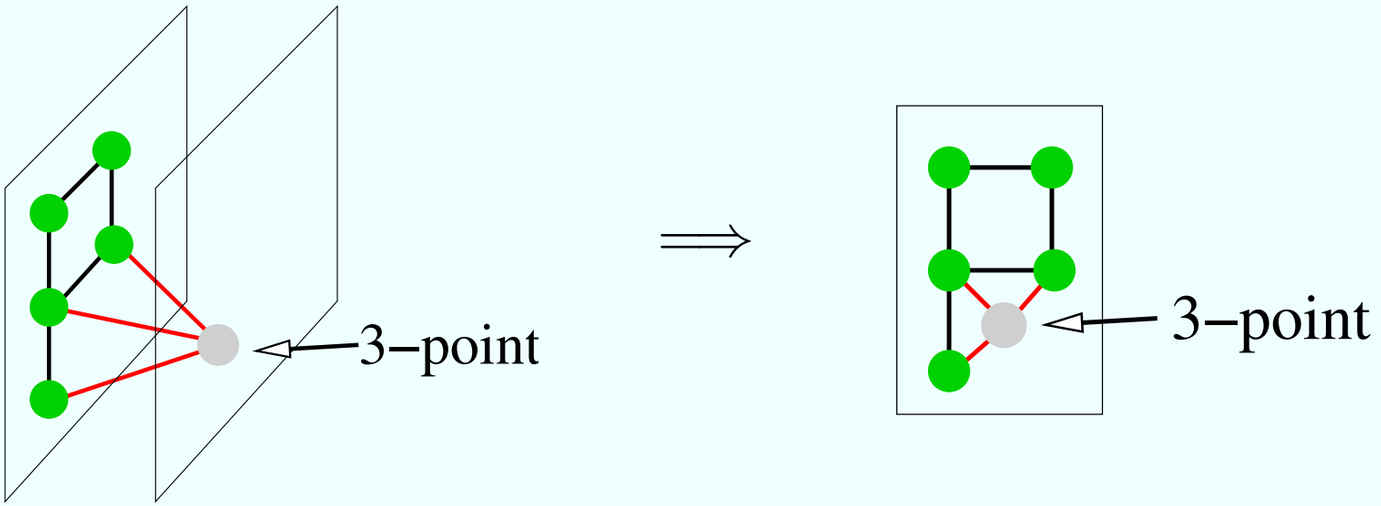
number of 4 $= n_i + 1 - a_i - b_i$ number of 2 $= 2a_i + 2b_i - 2\ell - 4$

number of 1 $= \ell + 4.$

for $\ell$, there is an upper bound [Backofen00]

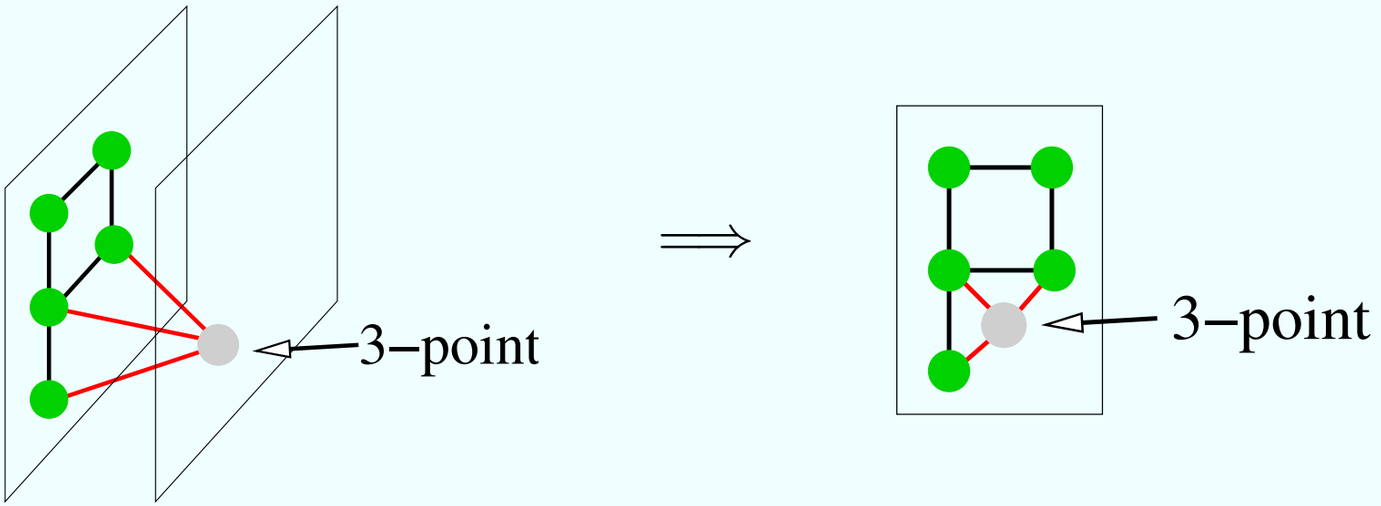
# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side ⟹ from the top

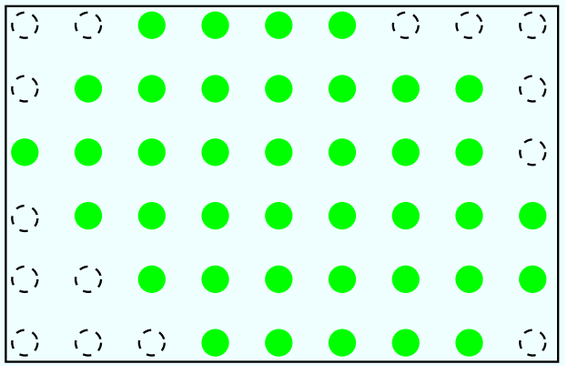


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top

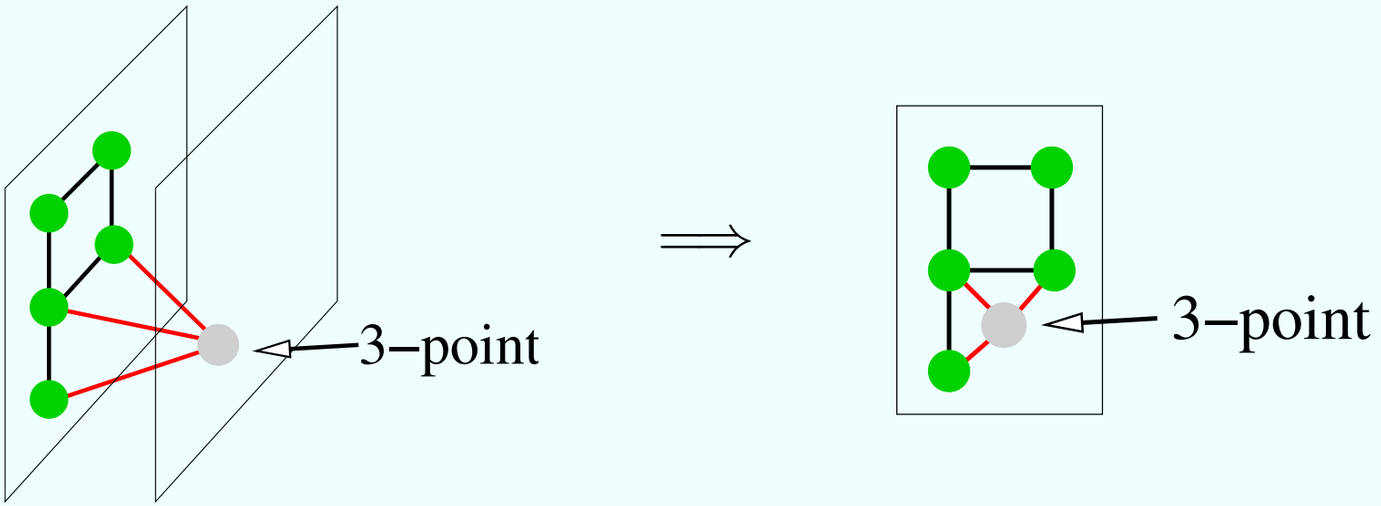



- observation: $\ell$ can also be calculated from the frame

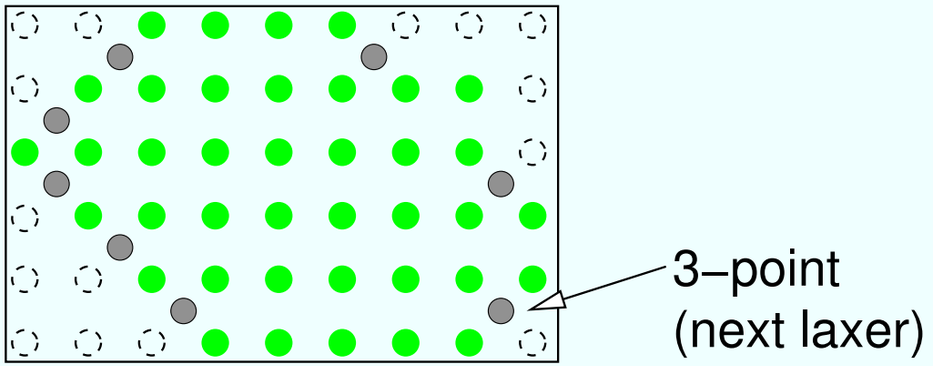


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top

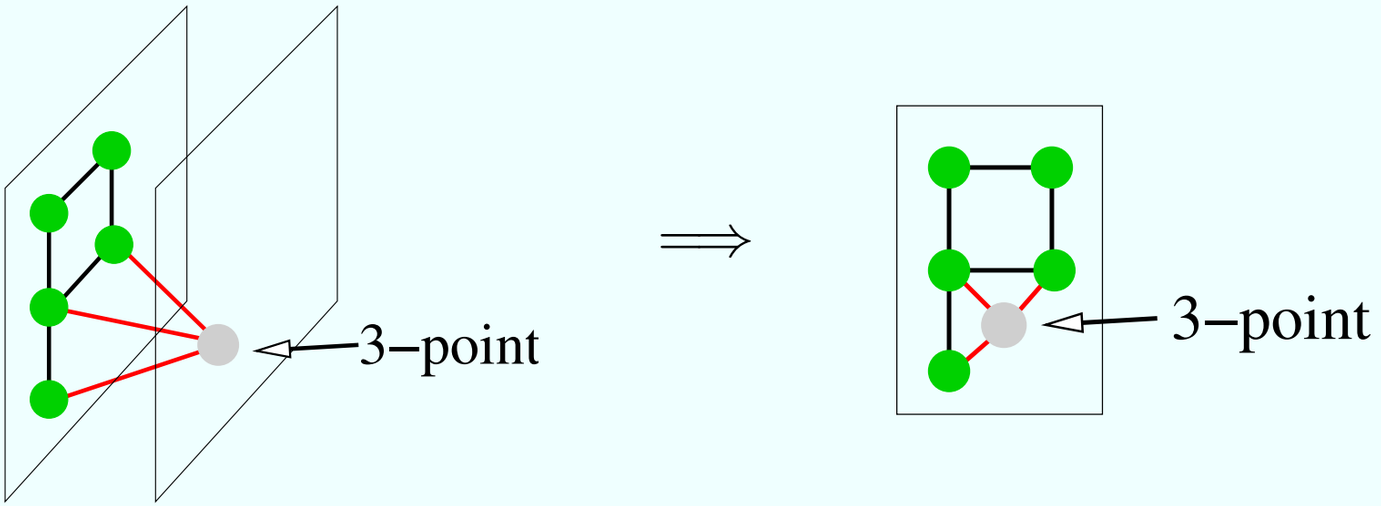



- observation: $\ell$ can also be calculated from the frame

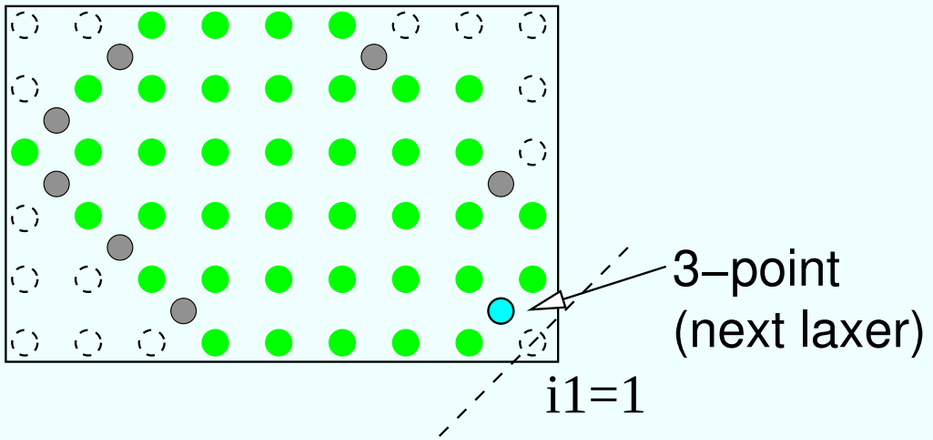


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top

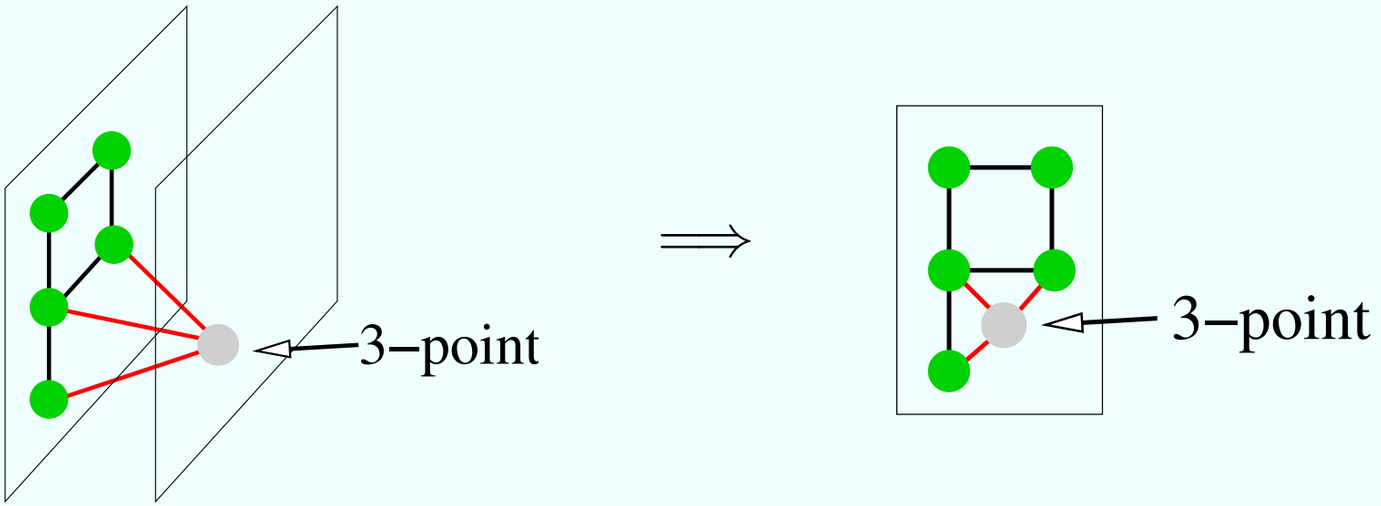



- observation: $\ell$ can also be calculated from the frame

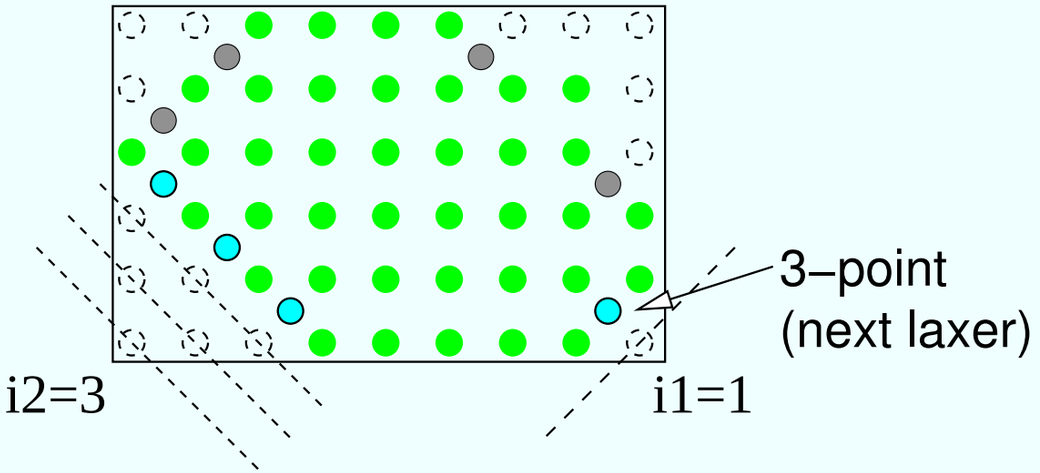


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top

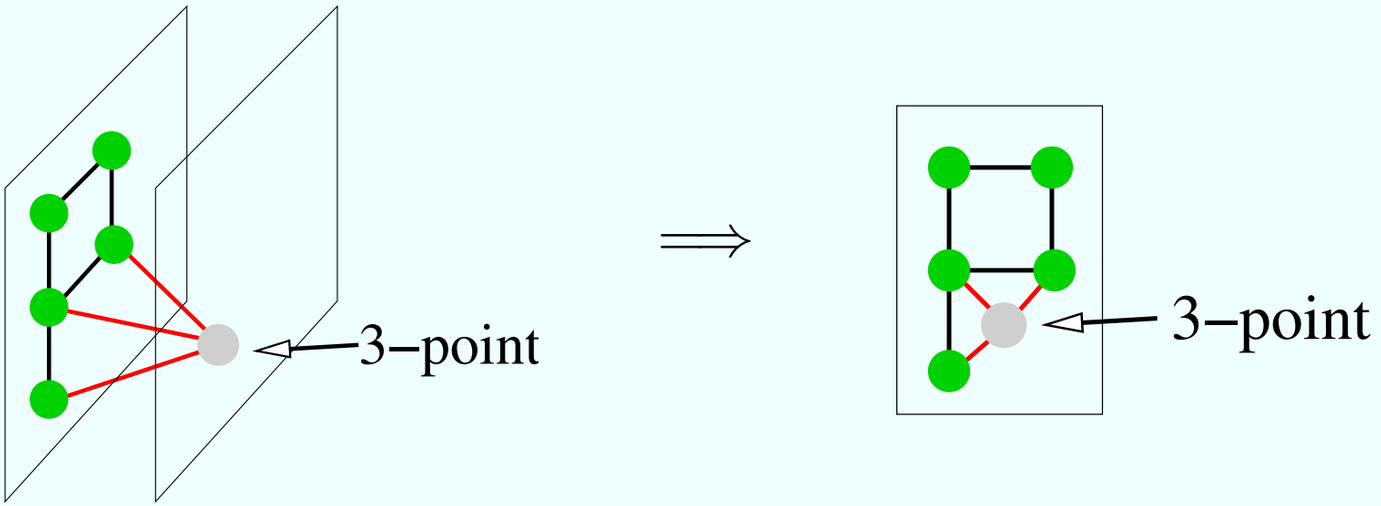



- observation: $\ell$ can also be calculated from the frame

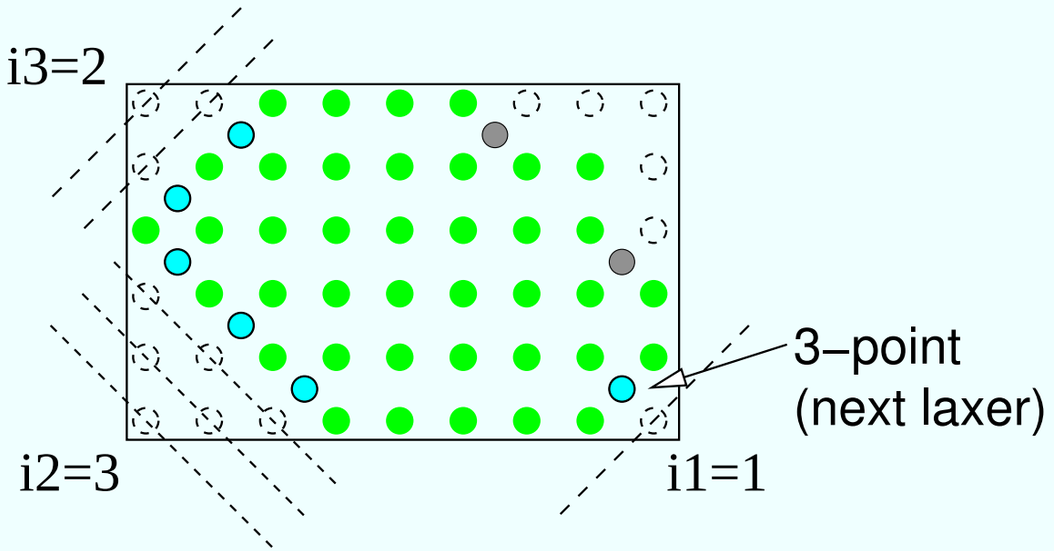


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top

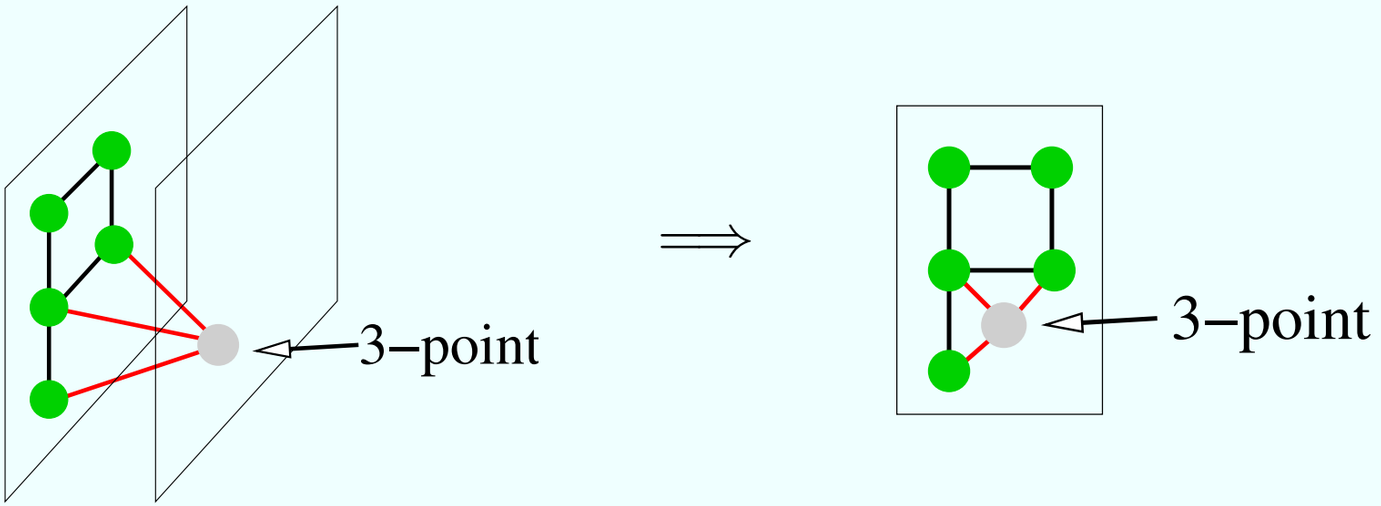



- observation: $\ell$ can also be calculated from the frame

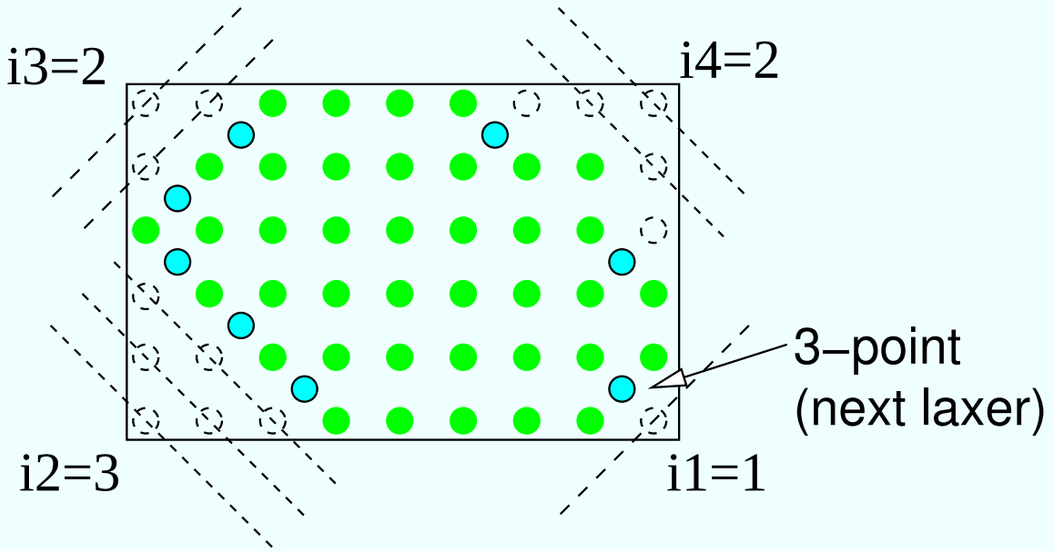


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top

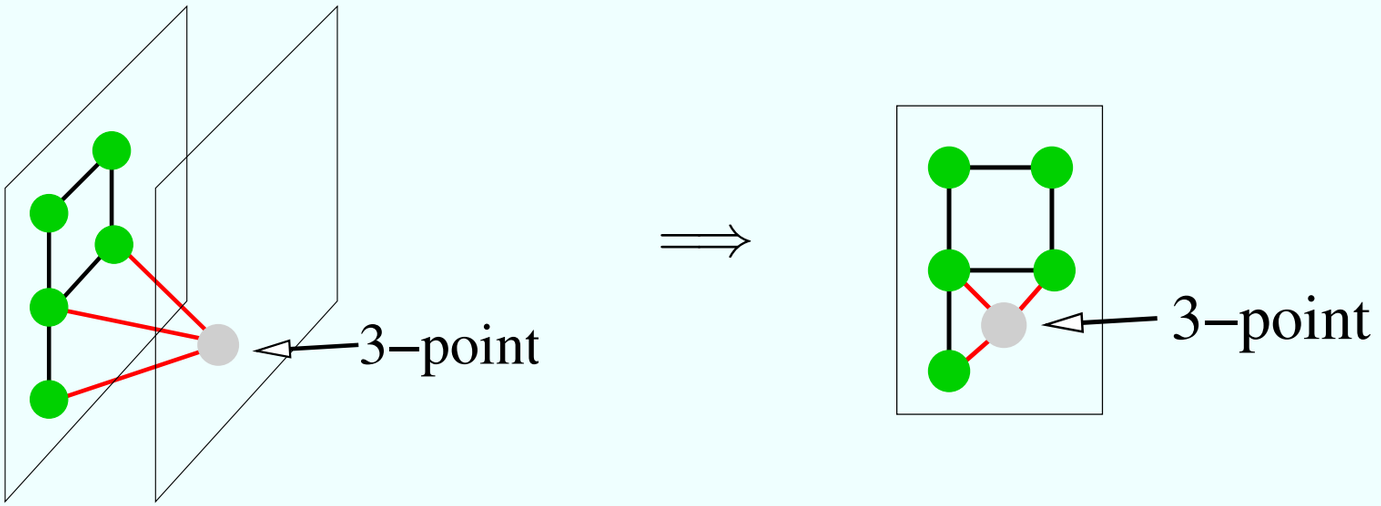



- observation: $\ell$ can also be calculated from the frame

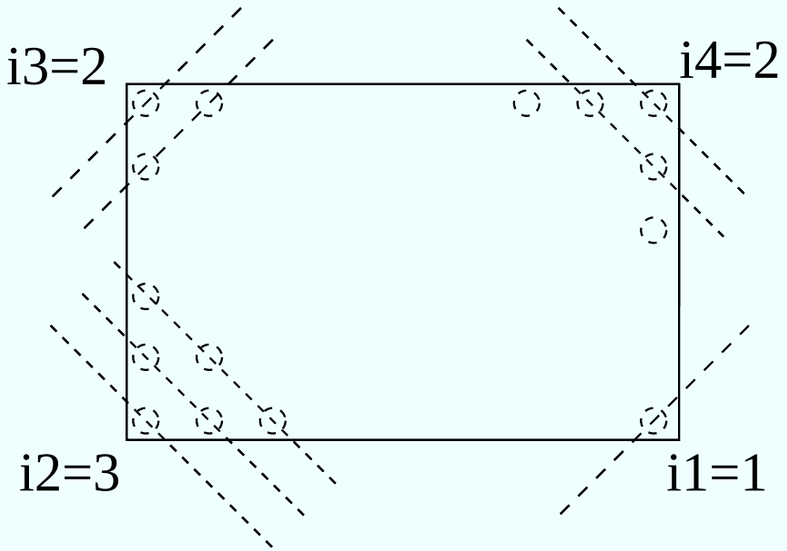


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top

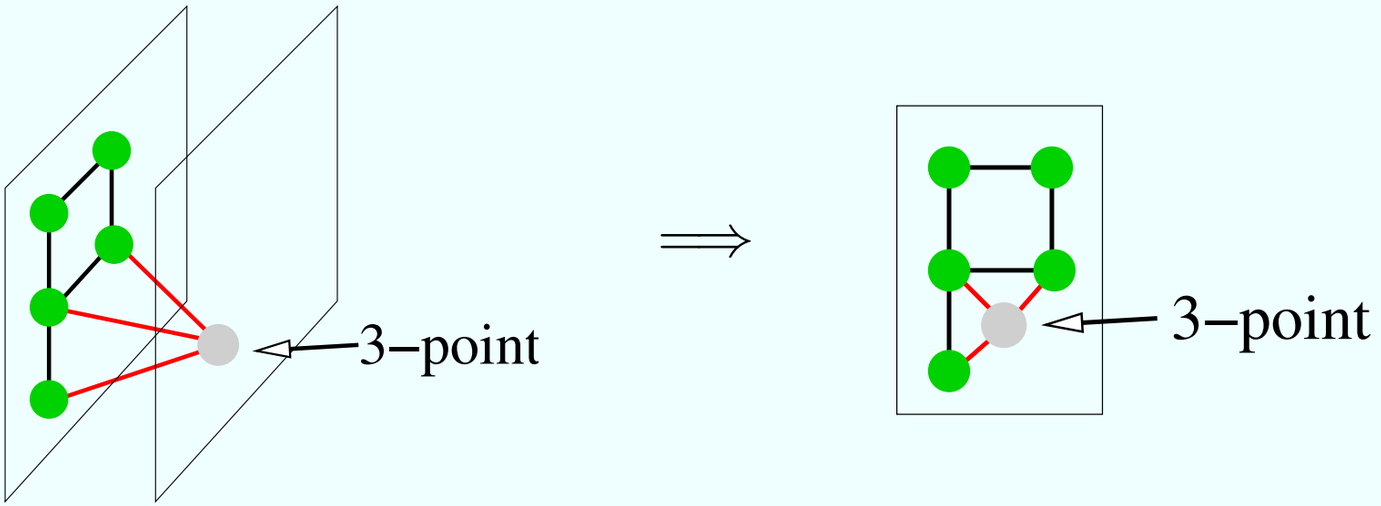



- observation: $\ell$ can also be calculated from the frame

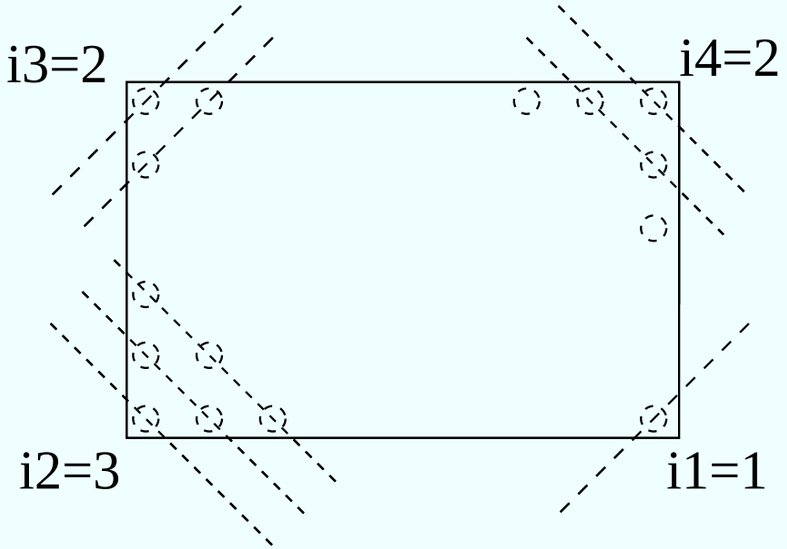


# Bounds on the Number $\ell$ of 3-Points

- **3-point:** from the side from the top



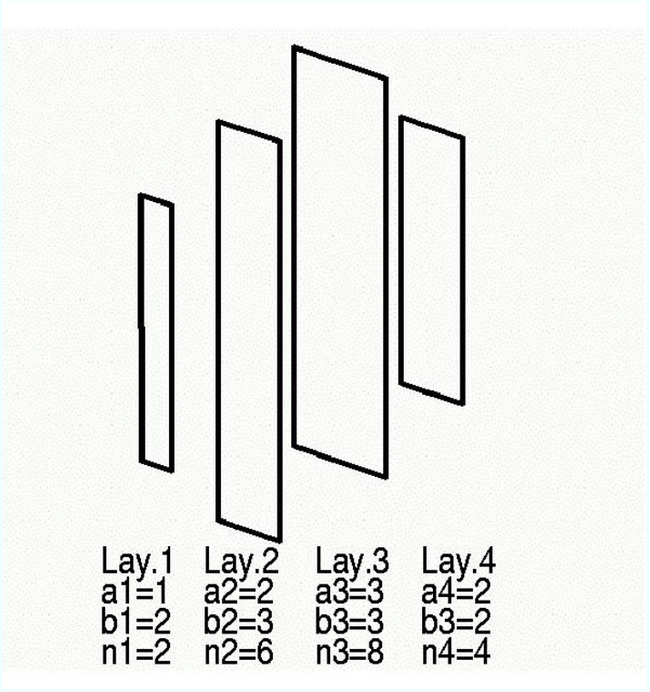

- observation: $\ell$ can also be calculated from the frame

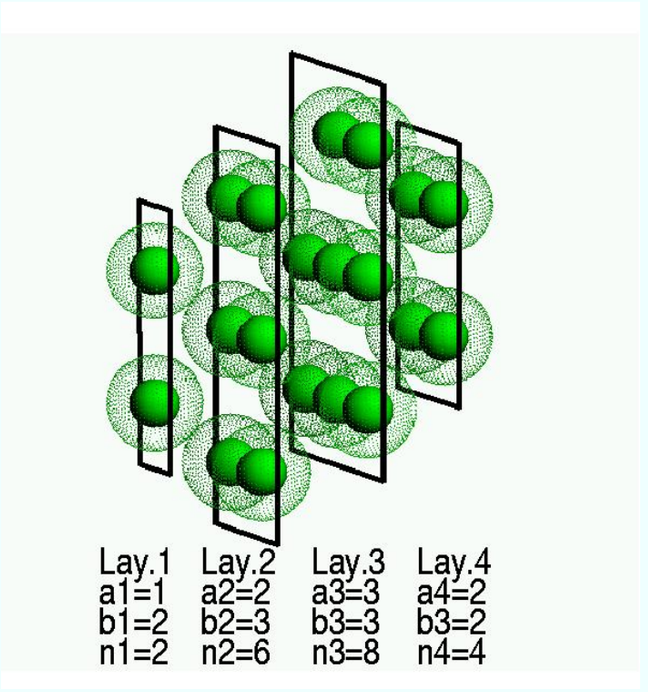



bound:

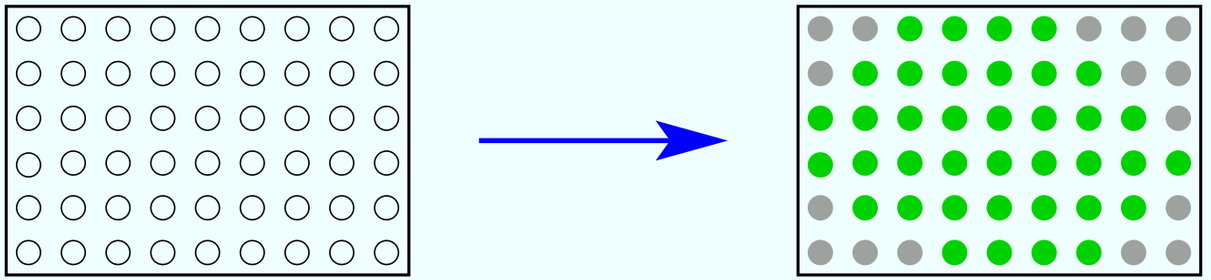
- calculate max. number of diagonals
- optimal placement: balance numbers between edges

## Problem 2: Enumerate Hydrophobic Cores



# Enumerating Hydrophobic Cores



- constraint variables:
  - boolean variable for every position


$\hat{=}$ $(\mathbf{pnt}(\vec{p}) = 1)$

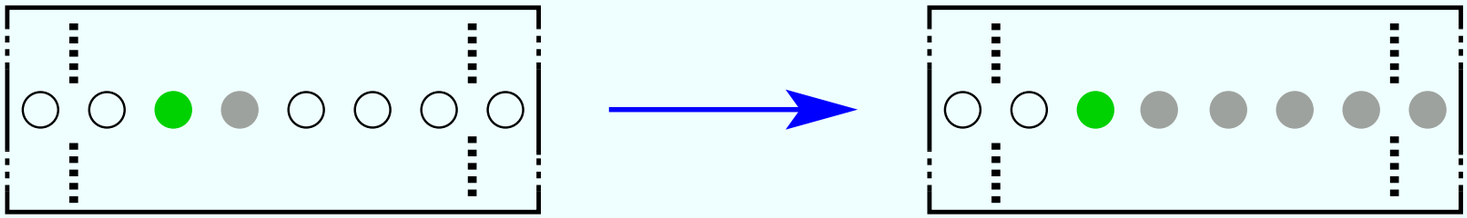

$\hat{=}$ $(\mathbf{pnt}(\vec{p}) = 0)$


$\hat{=}$ $(\mathbf{pnt}(\vec{p})$ undef.$)$

  - contact variable for each neighboring position


$\hat{=}$ $\mathbf{con}(\vec{p}, \vec{q}) = 1$

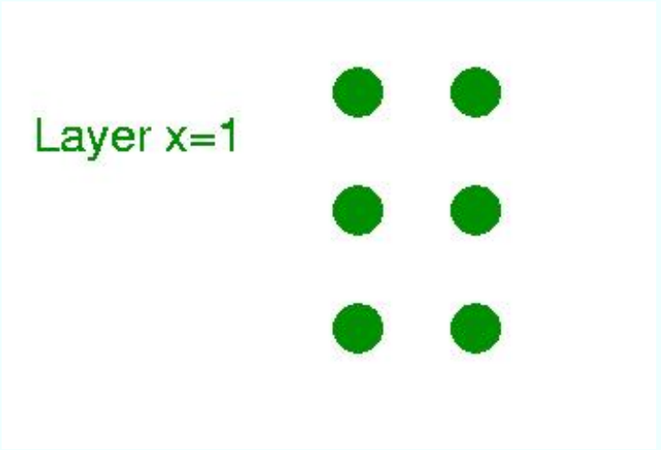
- constraints:
  - $\sum_{\vec{p} \in \mathrm{frames}} \mathbf{pnt}(\vec{p}) =$ number of Hs
  - if optimal, then no caveats



  - …

# Enumerating Hydrophobic Cores

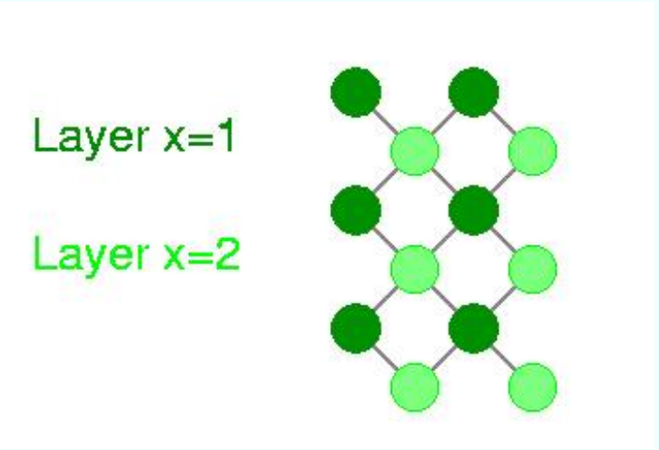
- remaining problem: relative positions of frames
- subproblems:
  - symmetries **later**



# Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

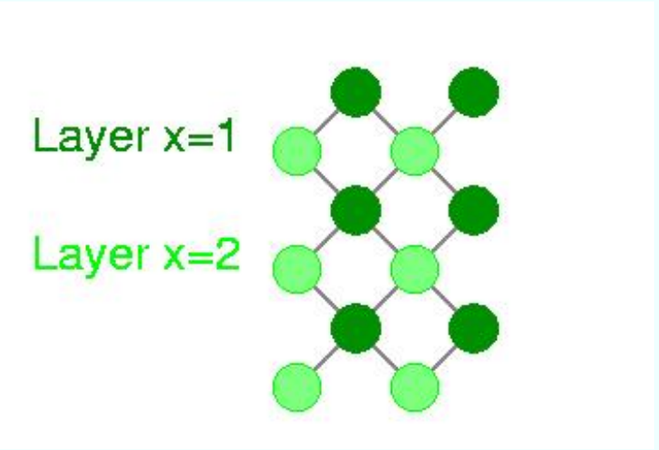
- subproblems:
  - symmetries **later**



# Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

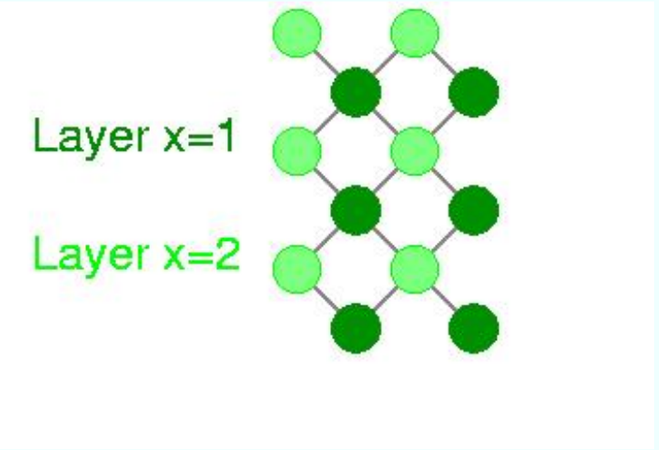
- subproblems:
  - symmetries **later**



# Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

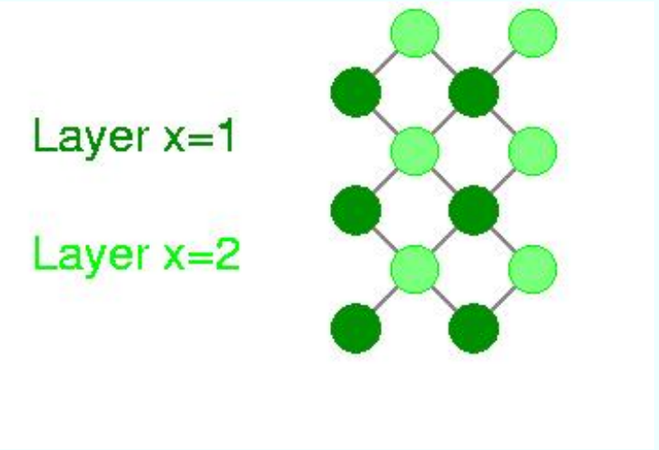
- subproblems:
  - symmetries **later**



# Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

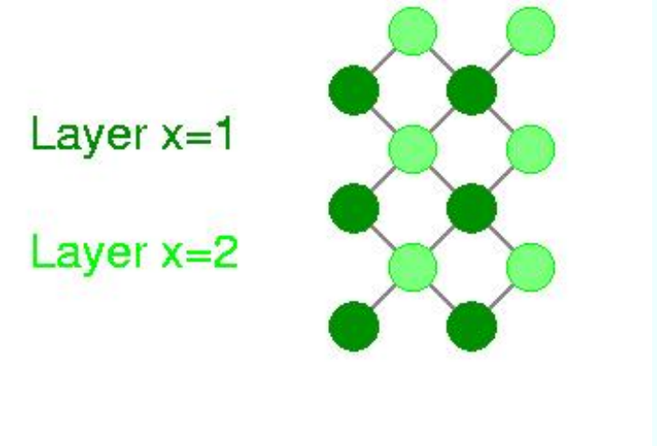
- subproblems:
  - symmetries **later**



# Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

- subproblems:
  - symmetries **later**

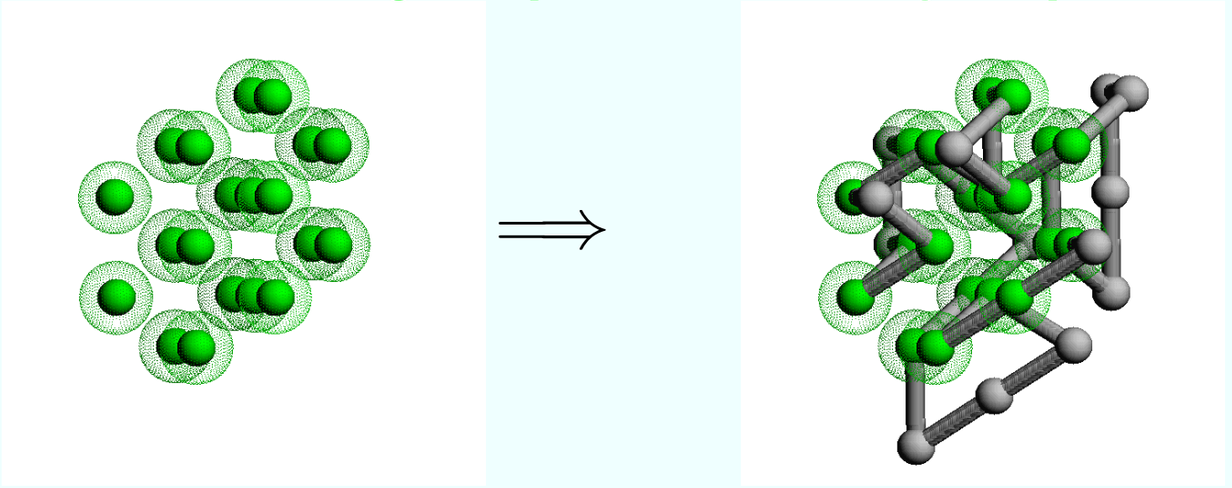



  - many subproblems solved several times
    - ∗ do not use fixed frame position
    - ∗ global bind frame positions by surrounding cube

  - more pruning: optimal core must have optimal frame-sequence in any direction

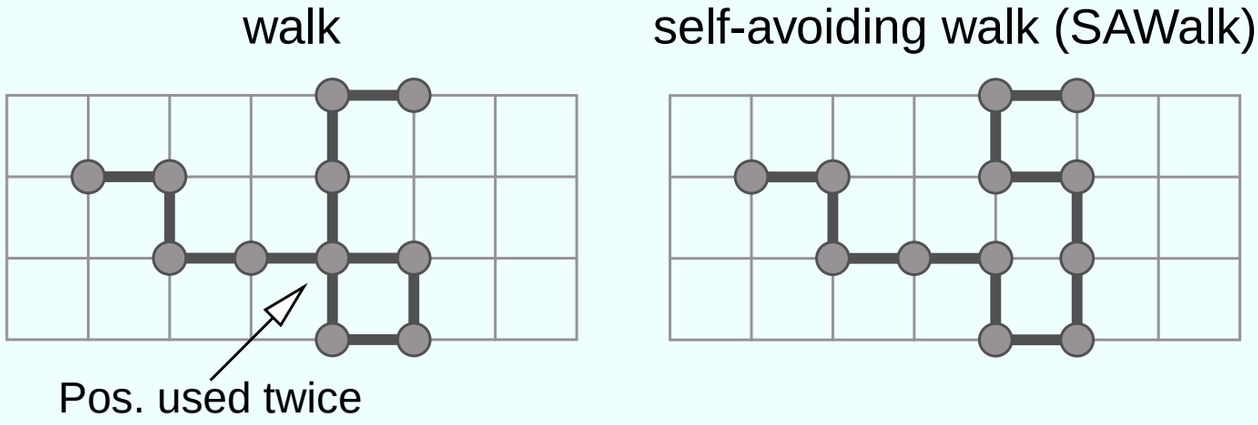
    **constructive disjunction**

## Problem 3: Threading Sequence onto Hydrophobic Cores



# New Constraints for Threading

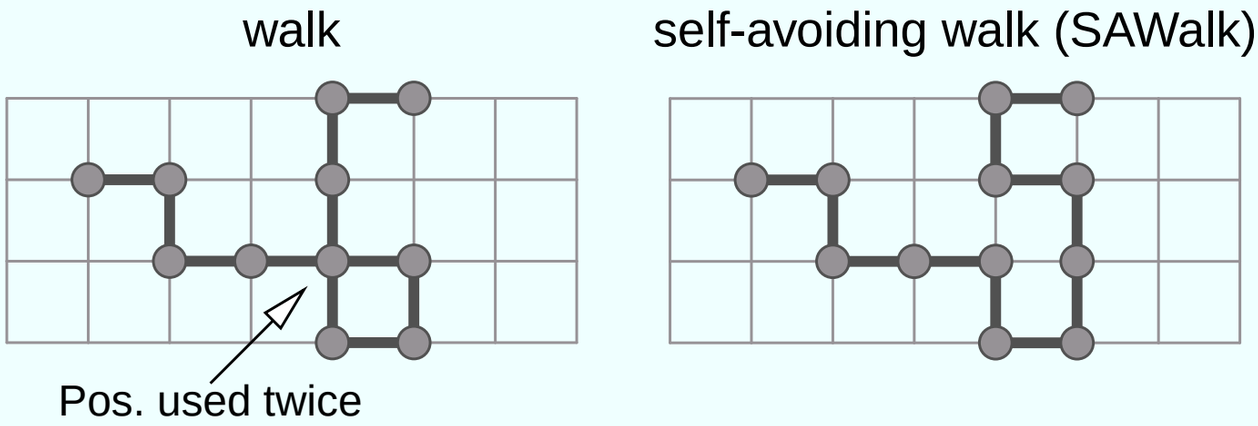
- threading: given core, find a sequence of monomer through it
- main problem: self-avoiding walks $\Rightarrow$ new constraint: $\texttt{SAWalk}(x_1, \ldots, x_m)$



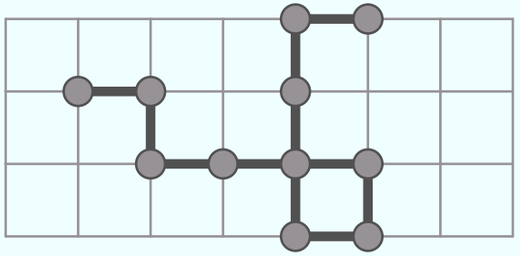

- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard

# New Constraints for Threading

- threading: given core, find a sequence of monomer through it
- main problem: self-avoiding walks $\Rightarrow$ new constraint: $\texttt{SAWalk}(x_1, \ldots, x_m)$

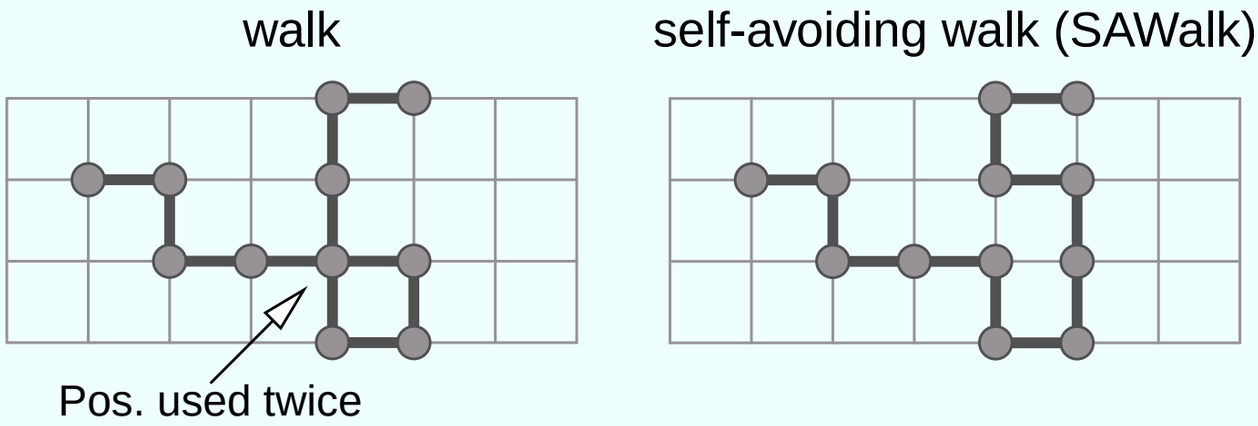



- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks $\Rightarrow$ k-avoiding walks

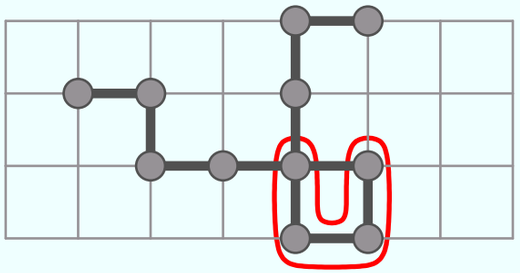


# New Constraints for Threading

- threading: given core, find a sequence of monomer through it
- main problem: self-avoiding walks $\Rightarrow$ new constraint: $\texttt{SAWalk}(x_1, \ldots, x_m)$

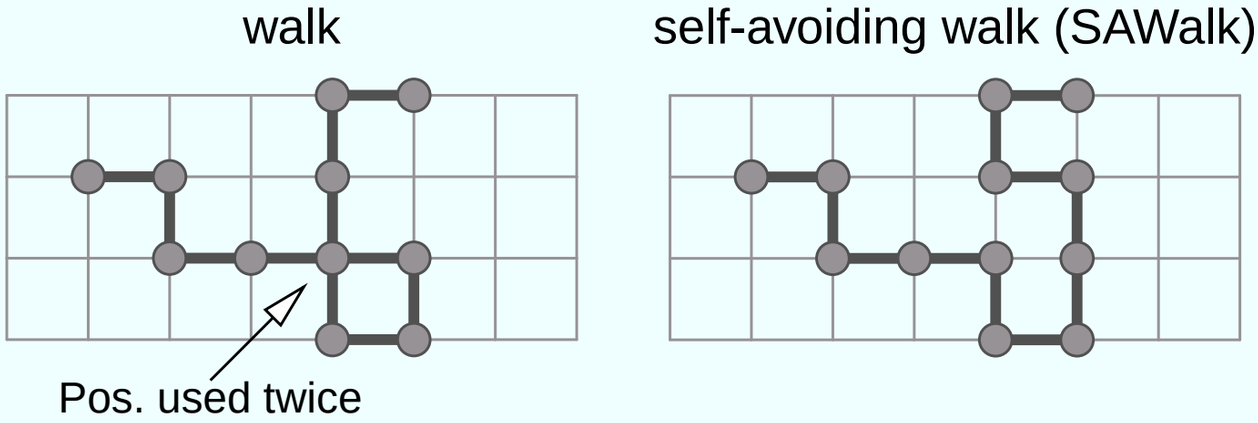



- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks $\Rightarrow$ k-avoiding walks

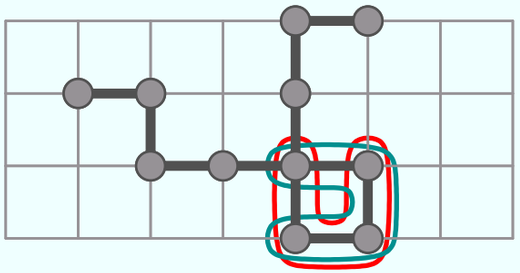


# New Constraints for Threading

- threading: given core, find a sequence of monomer through it
- main problem: self-avoiding walks $\Rightarrow$ new constraint: $\texttt{SAWalk}(x_1, \ldots, x_m)$



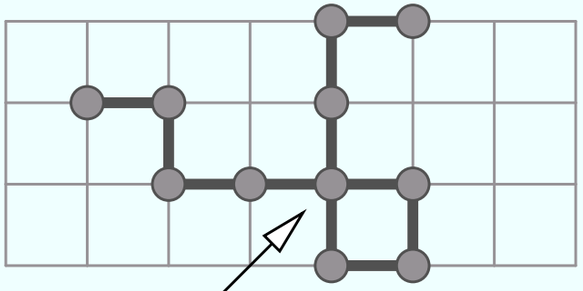

- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks $\Rightarrow$ k-avoiding walks



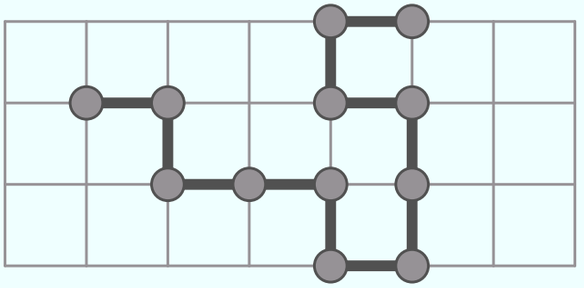
# New Constraints for Threading

- threading: given core, find a sequence of monomer through it
- main problem: self-avoiding walks $\Rightarrow$ new constraint: $\texttt{SAWalk}(x_1, \ldots, x_m)$

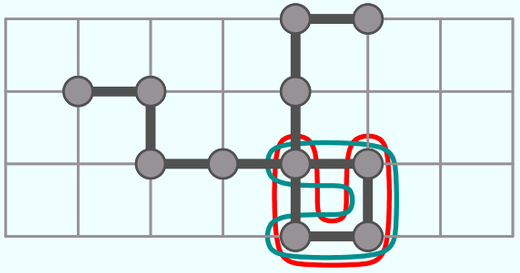
walk




self-avoiding walk (SAWalk)



- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks $\Rightarrow$ k-avoiding walks

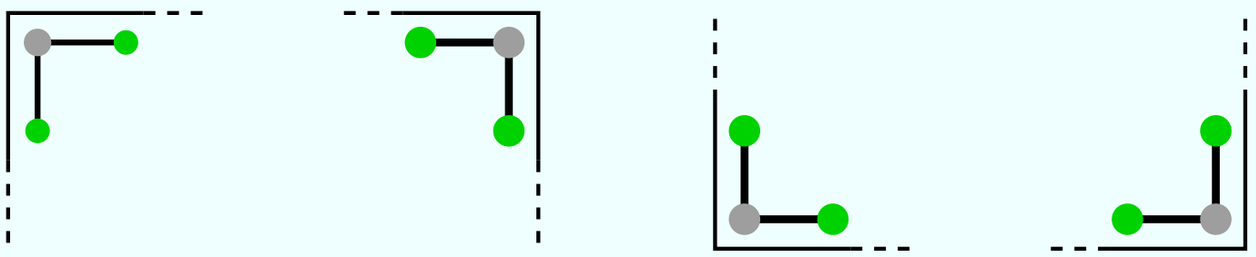


4- but not 5-avoiding

# Example: 3-Avoiding

- psinglets: HPH-subsequence

- in cubic lattice: has strong influence on core

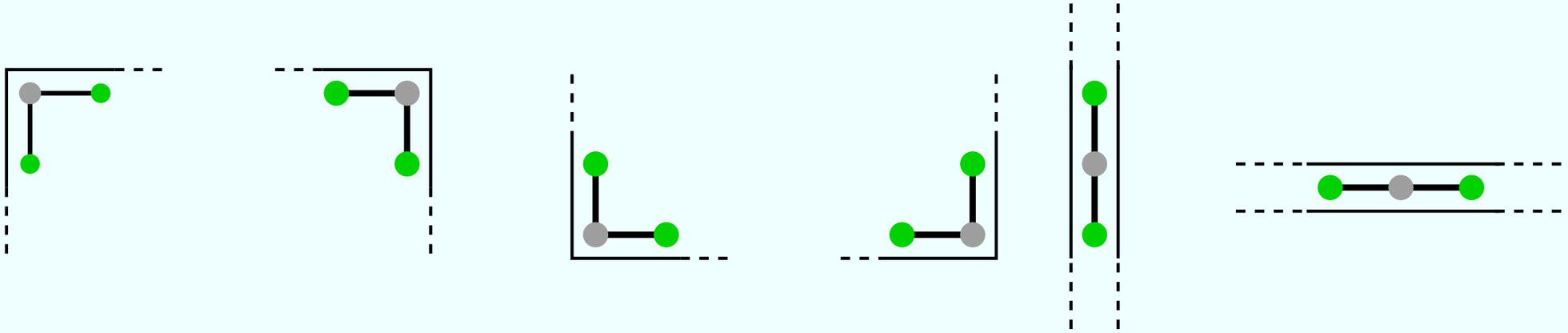


- caveat-freeness by path constraint

# Example: 3-Avoiding

- psinglets: HPH-subsequence

- in cubic lattice: has strong influence on core

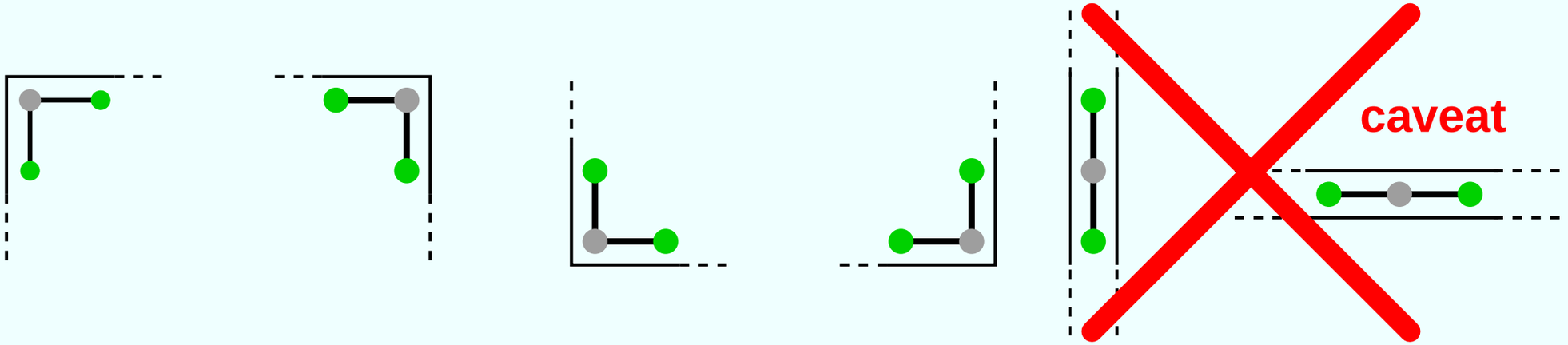


- caveat-freeness by path constraint

# Example: 3-Avoiding

- psinglets: HPH-subsequence

- in cubic lattice: has strong influence on core

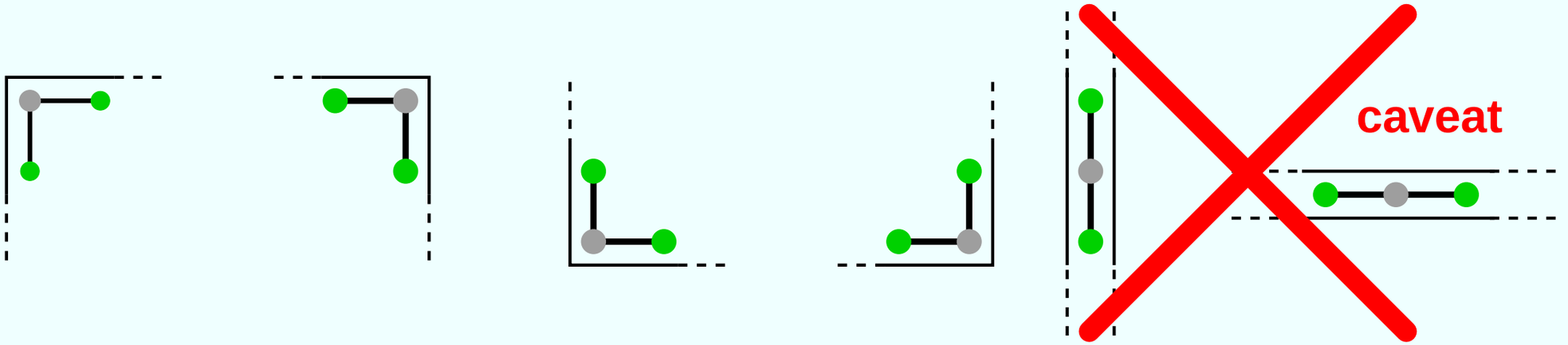



- caveat-freeness by path constraint

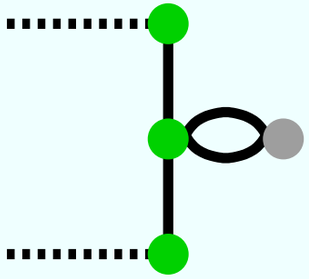
# Example: 3-Avoiding

- psinglets: HPH-subsequence

- in cubic lattice: has strong influence on core



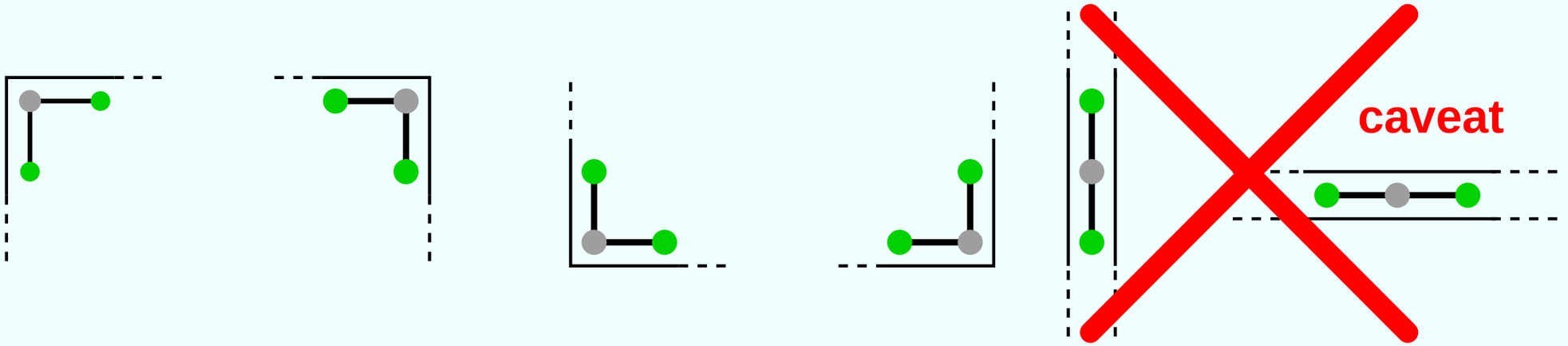

- caveat-freeness by path constraint
- remaining invalid case



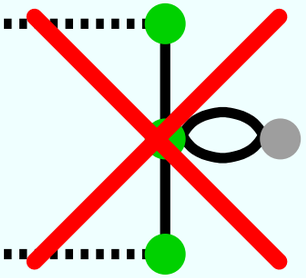
# Example: 3-Avoiding

- psinglets: HPH-subsequence

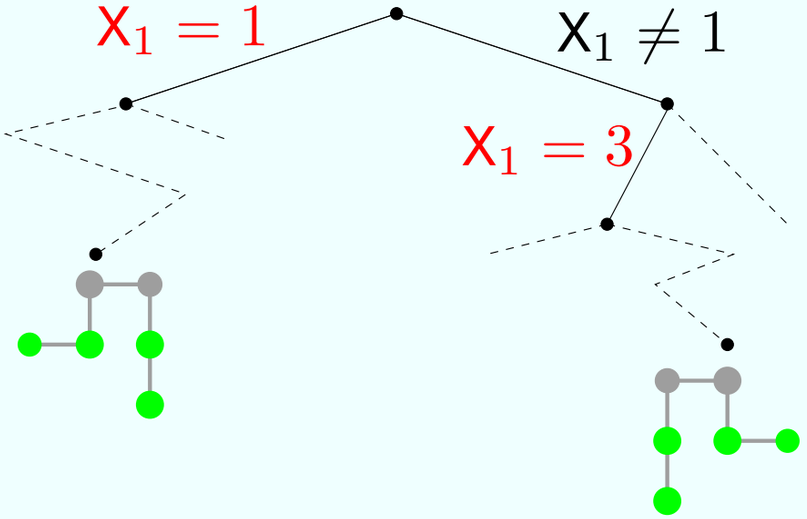
- in cubic lattice: has strong influence on core




- caveat-freeness by path constraint
- remaining invalid case **excluded by 3-avoidingness**



## Problem 4: Symmetry Breaking




**solved, but skipped here!**

# Comparison of Results

- small selection of previous approaches:

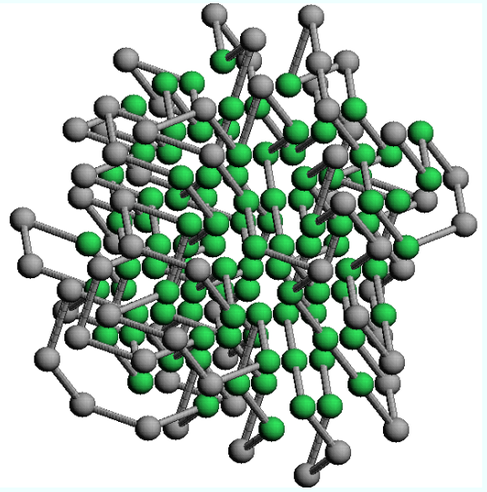
| authors | model | dim. | maxlen | algorithm | comment |
|---|---|---|---|---|---|
| [Yue& Dill PhysRevE93] | cubic HP | 3 | 36 | branch-and-bound | optimality proven |
| [Yue&Dill PNAS95] | cubic HP | 3 | 88 | branch-and-bound | optimality proven |
| [Sazhin et al. 01] | cubic HP, FCC | 3 | 34 | branch-and-bound | not always optimal |
| [Cui et al. PNAS02] | square HP | 2 | 18 | compl. enum | |
| [Hart&Istrail JCB97] | FCC side chain | 3 | — | approximation | $86\%$ of optimum |
| [Agarwala et al. JMB97] | FCC HP | 3 | — | approximation | $\frac{3}{5}$ of optimum |

# Comparison of Results

- small selection of previous approaches:

| authors | model | dim. | maxlen | algorithm | comment |
|---|---|---|---|---|---|
| [Yue& Dill PhysRevE93] | cubic HP | 3 | 36 | branch-and-bound | optimality proven |
| [Yue&Dill PNAS95] | cubic HP | 3 | 88 | branch-and-bound | optimality proven |
| [Sazhin et al. 01] | cubic HP, FCC | 3 | 34 | branch-and-bound | not always optimal |
| [Cui et al. PNAS02] | square HP | 2 | 18 | compl. enum | |
| [Hart&Istrail JCB97] | FCC side chain | 3 | — | approximation | $86\%$ of optimum |
| [Agarwala et al. JMB97] | FCC HP | 3 | — | approximation | $\frac{3}{5}$ of optimum |

- our results:
  - native conformation up to length 200
  - proof of optimality



threading on 100-Hs core

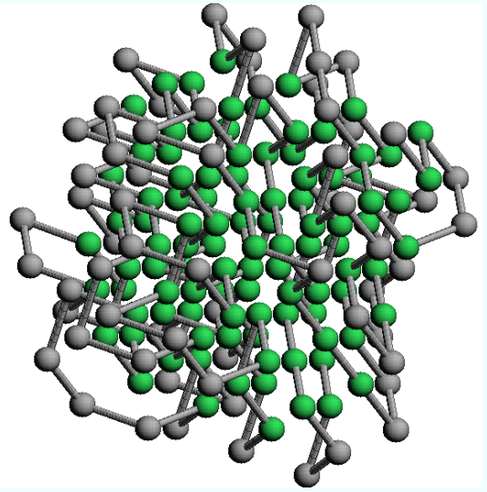
| seq. | length | runtime |
|---|---|---|
| S1 | 135 | 9 s |
| S2 | 151 | 15 s |
| S3 | 161 | 18 s |
| S4 | 164 | 11 s |

# Comparison of Results

- small selection of previous approaches:

| authors | model | dim. | maxlen | algorithm | comment |
|---|---|---|---|---|---|
| [Yue& Dill PhysRevE93] | cubic HP | 3 | 36 | branch-and-bound | optimality proven |
| [Yue&Dill PNAS95] | cubic HP | 3 | 88 | branch-and-bound | optimality proven |
| [Sazhin et al. 01] | cubic HP, FCC | 3 | 34 | branch-and-bound | not always optimal |
| [Cui et al. PNAS02] | square HP | 2 | 18 | compl. enum | |
| [Hart&Istrail JCB97] | FCC side chain | 3 | — | approximation | $86\%$ of optimum |
| [Agarwala et al. JMB97] | FCC HP | 3 | — | approximation | $\frac{3}{5}$ of optimum |

- our results:
  - native conformation up to length 200
  - proof of optimality
  - number of conformations of length n: $\approx 4.5^n$



threading on 100-Hs core

| seq. | length | runtime |
|---|---|---|
| S1 | 135 | 9 s |
| S2 | 151 | 15 s |
| S3 | 161 | 18 s |
| S4 | 164 | 11 s |

$\Rightarrow$ **search space handled** $\approx 4.5^{90}$ **bigger**

  - only existing non-heuristic algorithm for **FCC**

# Comparison of Results

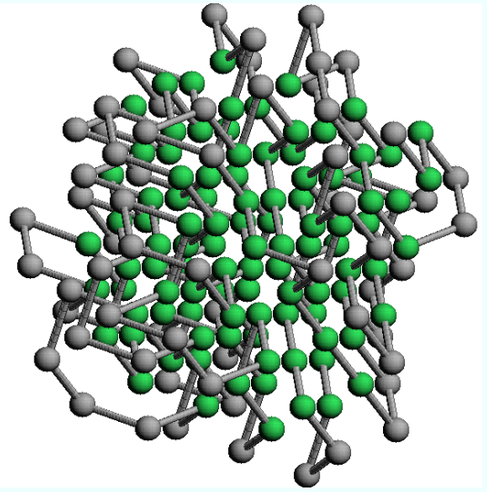
- small selection of previous approaches:

| authors | model | dim. | maxlen | algorithm | comment |
|---|---|---|---|---|---|
| [Yue& Dill PhysRevE93] | cubic HP | 3 | 36 | branch-and-bound | optimality proven |
| [Yue&Dill PNAS95] | cubic HP | 3 | 88 | branch-and-bound | optimality proven |
| [Sazhin et al. 01] | cubic HP, FCC | 3 | 34 | branch-and-bound | not always optimal |
| [Cui et al. PNAS02] | square HP | 2 | 18 | compl. enum | |
| [Hart&Istrail JCB97] | FCC side chain | 3 | — | approximation | $86\%$ of optimum |
| [Agarwala et al. JMB97] | FCC HP | 3 | — | approximation | $\frac{3}{5}$ of optimum |

- our results:
  - native conformation up to length **300**
  - proof of optimality
  - number of conformations of length n: $\approx 4.5^n$

$\Rightarrow$ **search space handled** $\approx 4.5^{190}$ **bigger**

  - only existing non-heuristic algorithm for **FCC**



threading on 100-Hs core

| seq. | length | runtime |
|---|---|---|
| S1 | 135 | 9 s |
| S2 | 151 | 15 s |
| S3 | 161 | 18 s |
| S4 | 164 | 11 s |

# Runtimes

prediction of *one* optimal structure

(sequence length 48, “Harvard sequences” from [Yue *et al.*, 1995])

| Nr. | sequence | CPSP | PERM |
|---|---|---|---|
| 1 | $HPH_2P_2H_4PH_3P_2H_2P_2HPH_3PHPH_2P_2H_2P_3HP_8H_2$ | 0,1 s | 6,9 min |
| 2 | $H_4PH_2PH_5P_2HP_2H_2P_2HP_6HP_2HP_3HP_2H_2P_2H_3PH$ | 0,1 s | 40,5 min |
| 3 | $PHPH_2PH_6P_2HPHP_2HPH_2PHPHP_3HP_2H_2P_2H_2P_2HPHP_2HP$ | 4,5 s | 100,2 min |
| 4 | $P_2HP_3HPH_4P_2H_4PH_2PH_3P_2HPHPHP_2HP_6H_2PH_2PH$ | 1,8 s | 74,7 min |
| 5 | $H_3P_3H_2PHPH_2PH_2PH_2PHP_7HPHP_2HP_3HP_2H_6PH$ | 1,7 s | 59,2 min |
| 6 | $PHP_4HPH_3PHPH_4PH_2PH_2P_3HPHP_3H_3P_2H_2P_2H_2P_3H$ | 12,1 s | 144,7 min |
| 7 | $PHPH_2P_2HPH_3P_2H_2PH_2P_3H_5P_2HPH_2PHPHP_4HP_2HPHP$ | 7,3 s | 284,0 min |
| 8 | $PH_2PH_3PH_4P_2H_3P_6HPH_2P_2H_2PHP_3H_2PHPHPH_2P_3$ | 1,5 s | 26,6 min |
| 9 | $PHPHP_4HPHPHP_2HPH_6P_2H_3PHP_2HPH_2P_2HPH_3P_4H$ | 0,3 s | 1420,0 min |
| 10 | $PH_2P_6H_2P_3H_3PHP_2HPH_2P_2HP_2HP_2H_2P_2H_7P_2H_2$ | 0,1 s | 18,3 min |

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

  PERM=pruned-enriched Rosenbluth method

# Applications

- structure prediction

- investigation of landscape properties
  - degeneracy of sequences
  - finding protein-likes sequences with unique ground state
  - comparing different models (cubic/fcc, HP-model with HPNX)

# Degeneracy

- degeneracy (g) of a sequence = number of structures with lowest energy

- known: HP-model has high degeneracy
- unknow:
  - how high is it?
  - are there sequences with g=1 (unique ground state, “protein-like”)?
  - how does it compare to other models (FCC, HPNX)?
  - how do neutral nets look like?

# Degeneracy

- degeneracy (g) of a sequence = number of structures with lowest energy

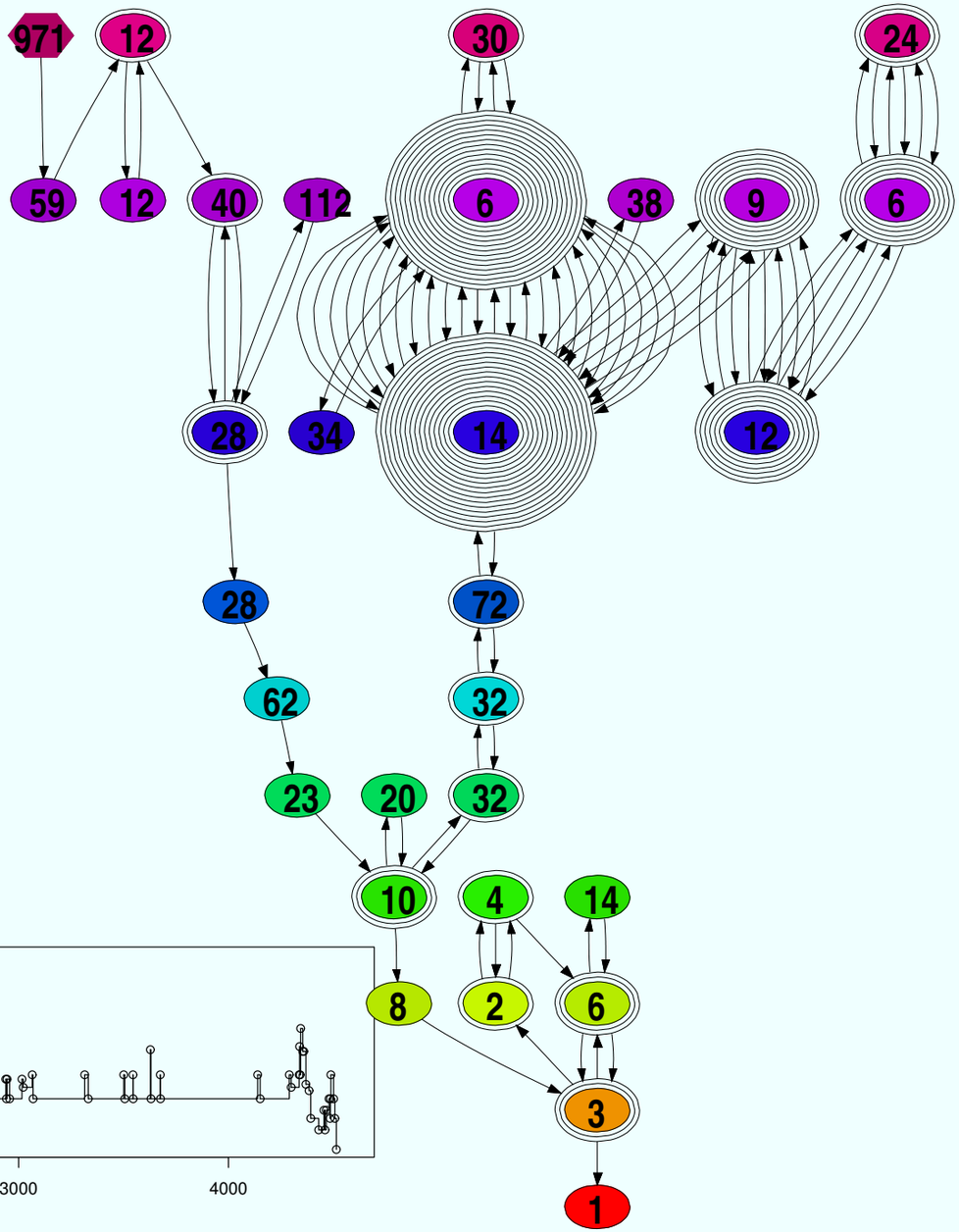
- known: HP-model has high degeneracy
- unknow:
  - how high is it?
  - are there sequences with g=1 (unique ground state, “protein-like”)?
  - how does it compare to other models (FCC, HPNX)?
  - how do neutral nets look like?

- degeneracy: can only be tested via two algothms

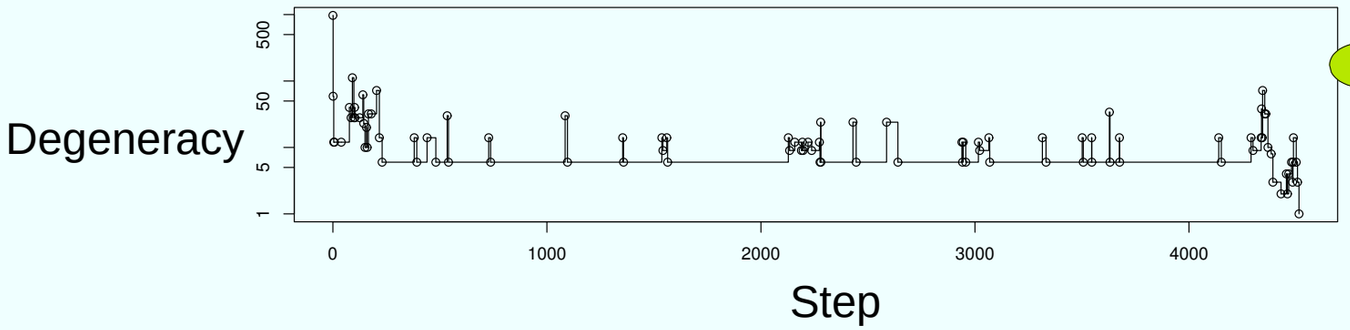
| Sequence | degeneracy found by | |
|---|---|---|
| | CHCC [Yue et al] | our approach |
| HPHHPPHHHHPHHHPPHHPPHPHHHPHPHHPPHHPPPHPPPPPPPPHH | $\geq 1,500,000$ | $10,677,113$ |
| HHHHPHHPHHHHHPPHPPHHPPHPPPPPPHPPHPPPHPPHHPPHHHPH | $\geq 14,000$ | $28,180$ |
| PHPHHPHHHHHHPPHPHPPHPHHPHPHPPPHPPHHPPHHPPHPHPPHP | $\geq 5,000$ | $5,090$ |
| PHHPPPPPPHHPPPHHHPHPPHPHHPPHPPHPPHHPPHHHHHHHPPHH | $\geq 188,000$ | $580,751$ |

# Application: Design of protein-like Sequences

- find sequences with *exactly* *one* optimal structure
- stochastic local search

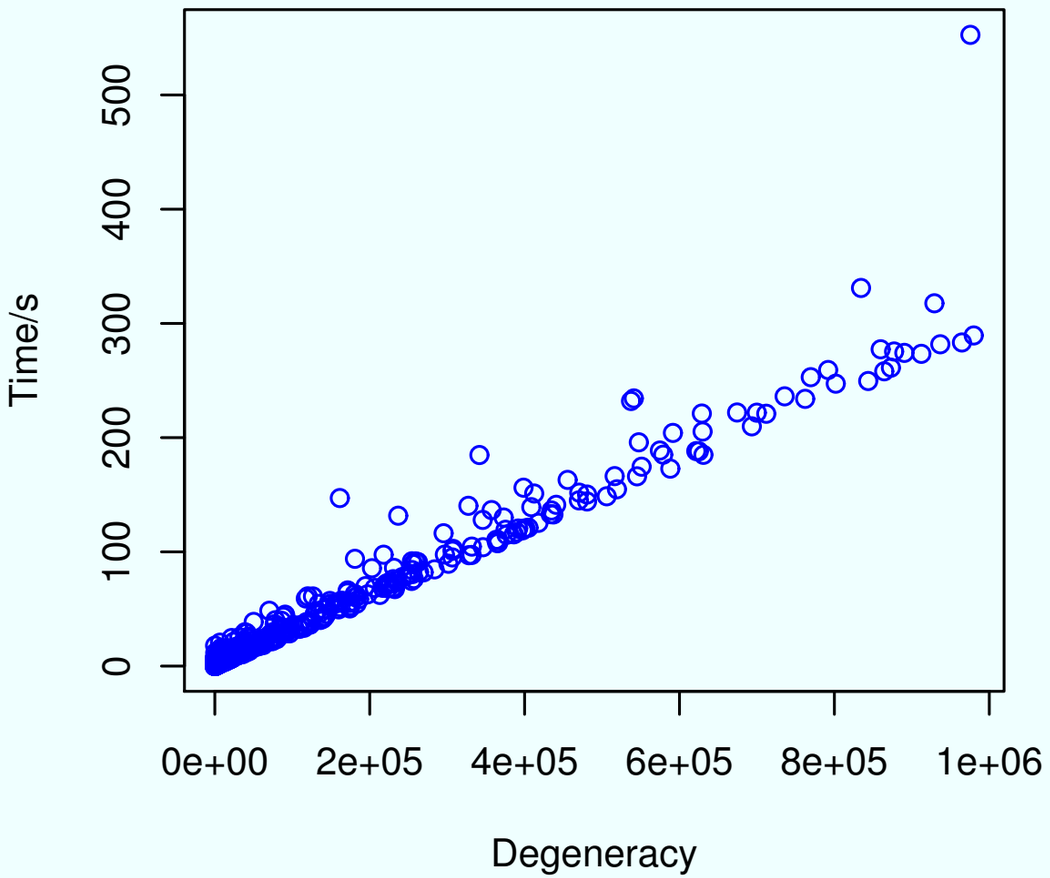



*node*:
accepted sequences
*edges*:
simulation step/mutation



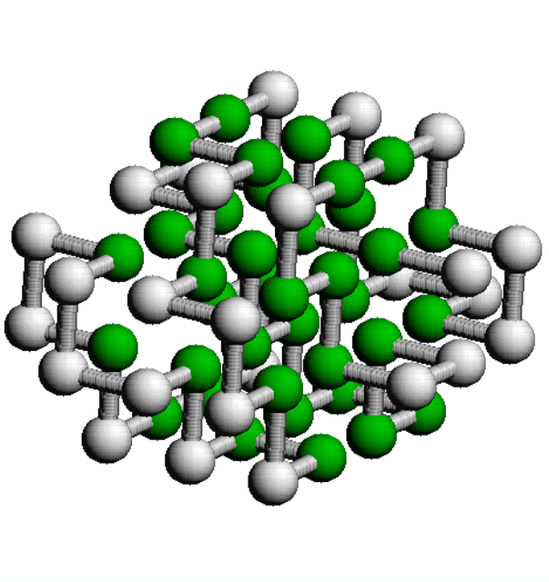
# Run Time Requirements

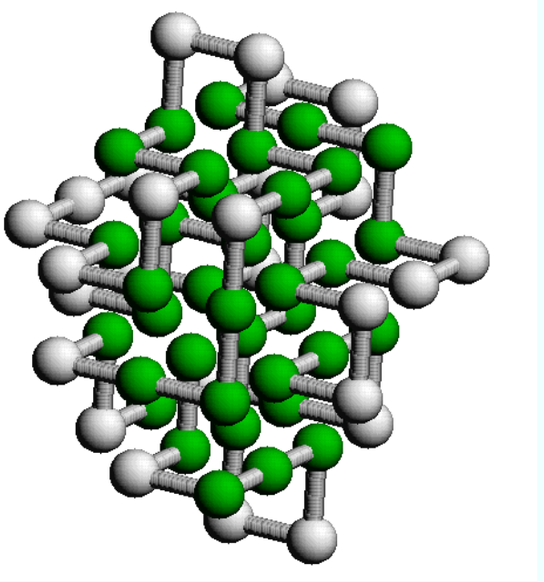
- at every step: calculation/estimation of degeneracy (using our CPFL)
- but: runtime depends on degeneracy
- good news: runtime grows only linearly with degeneracy



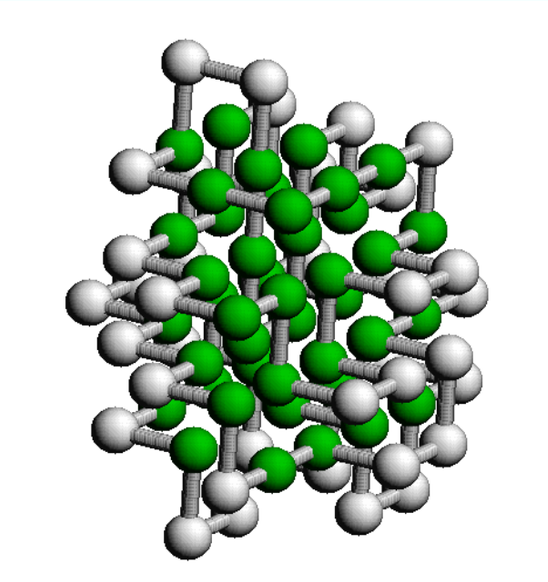
# Example: Sequences with Unique Ground-State

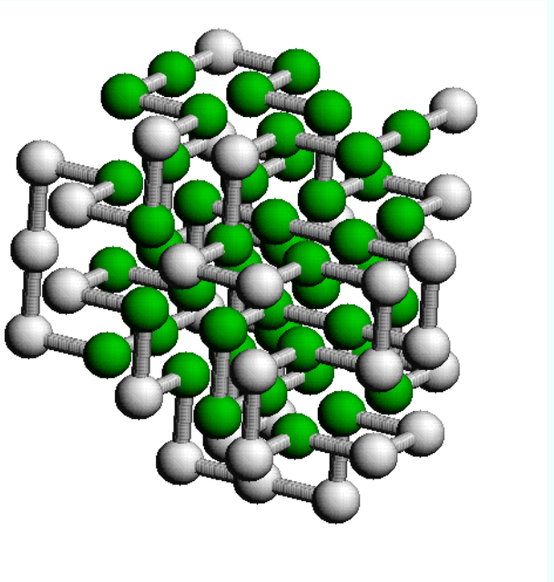
- length 64:





- length 80:

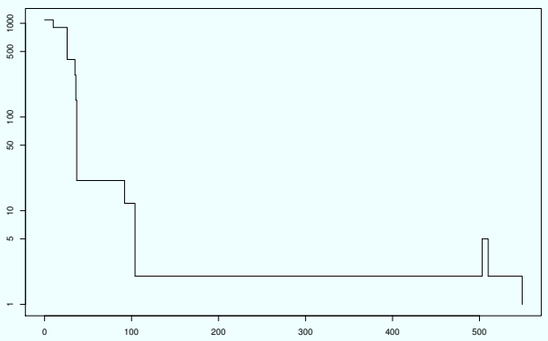




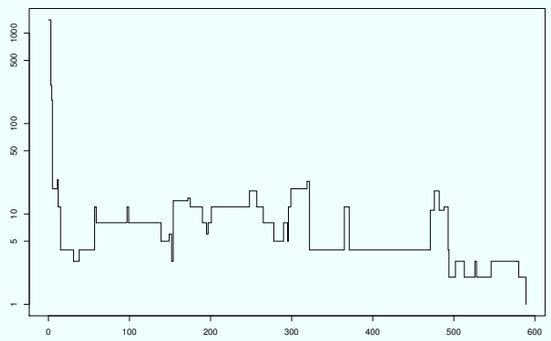
- *Note: previously it was assumed that HP-model has none g=1 sequences*

# Three “Typical” Runs

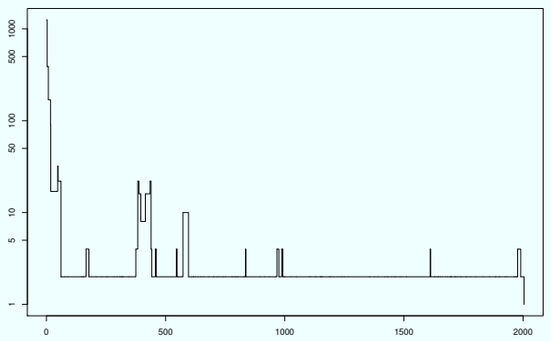
a)



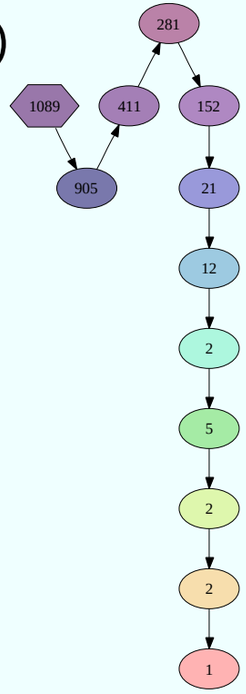

b)



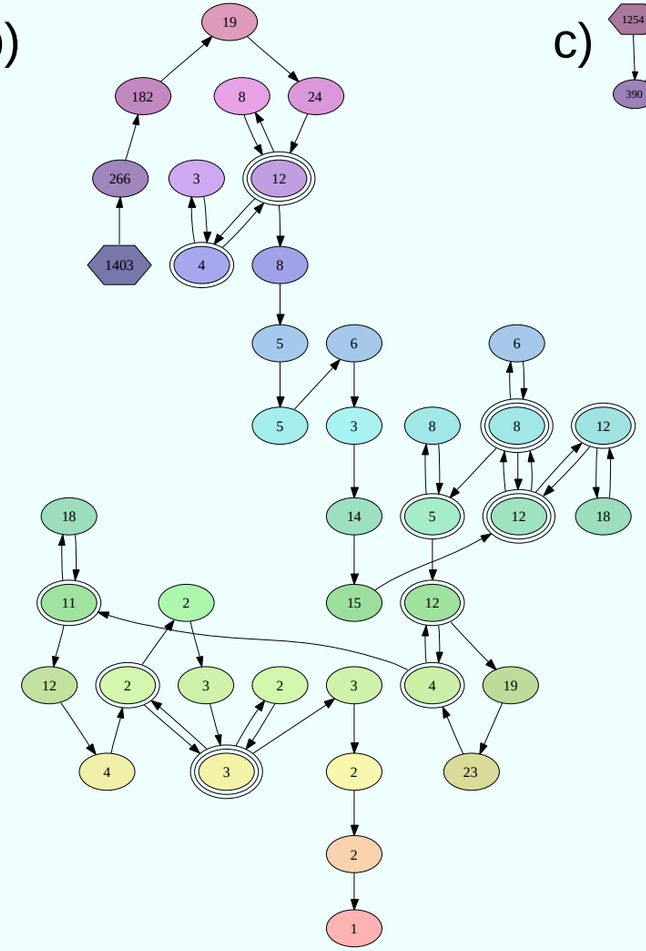

c)



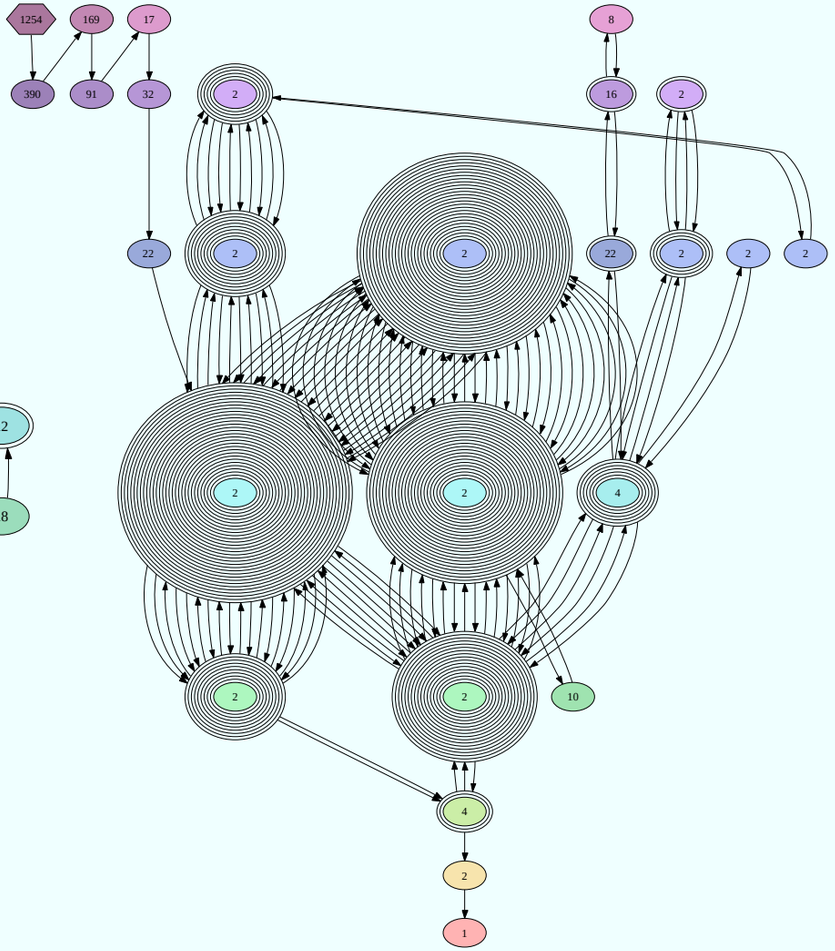

a)




b)

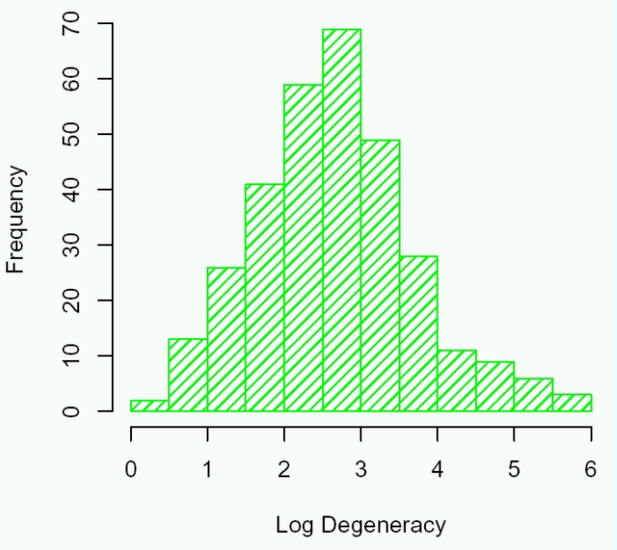



c)

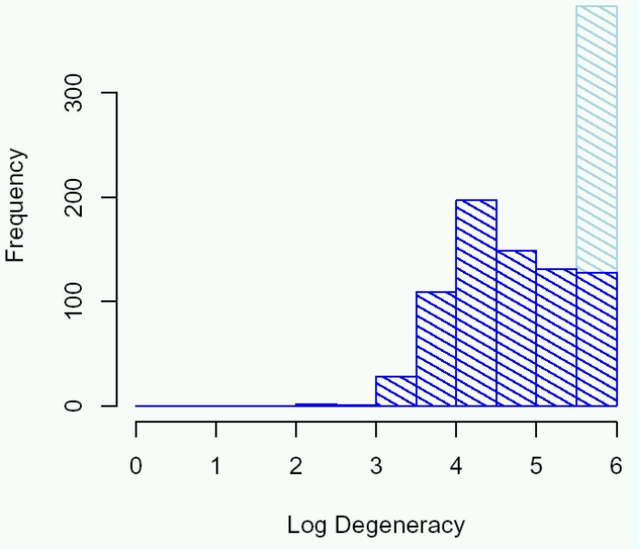


# Degeneracy: FCC vs. Cubic

- log-degeneracy cubic HP-model:




- log-degeneracy FCC HP-model:



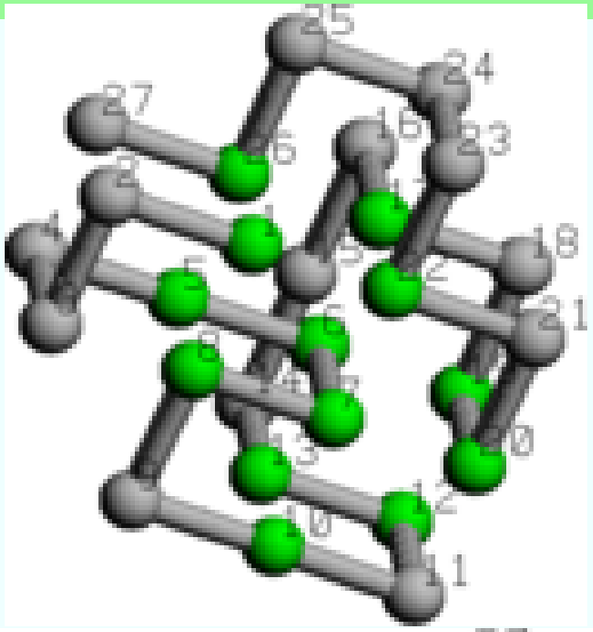
# Degeneracy: HP-Model vs. HPNX-model

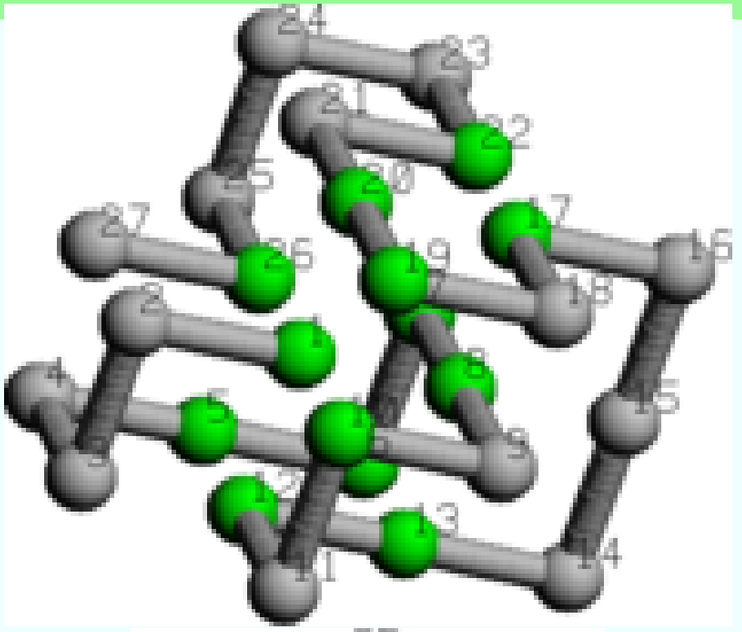
- HPNX: P=positive N=negative X=neutral
- should reduce the degeneracy

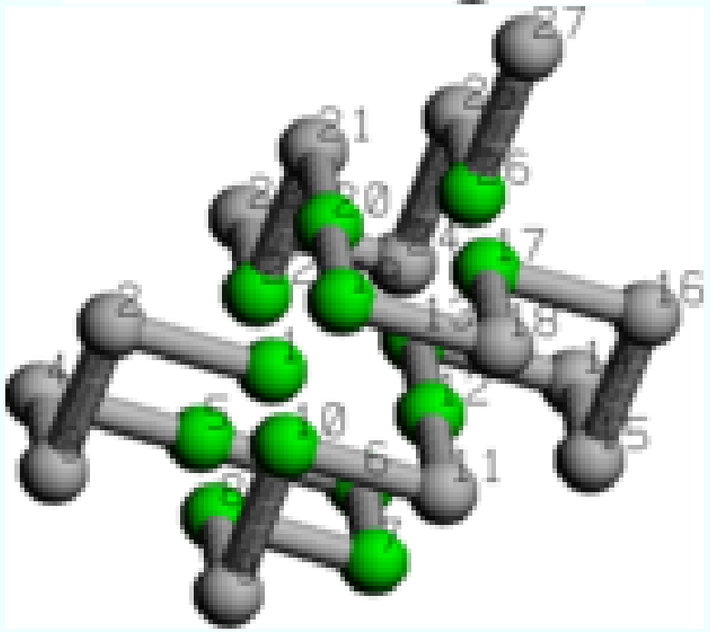
- How much? $\Rightarrow$ preliminary results
  - HP: approx. 0.016% of all random sequences are uniquely folding.
  - HPNX: approx. 2.6% of all random sequences are uniquely folding.
- *Note: 50% H monomers*

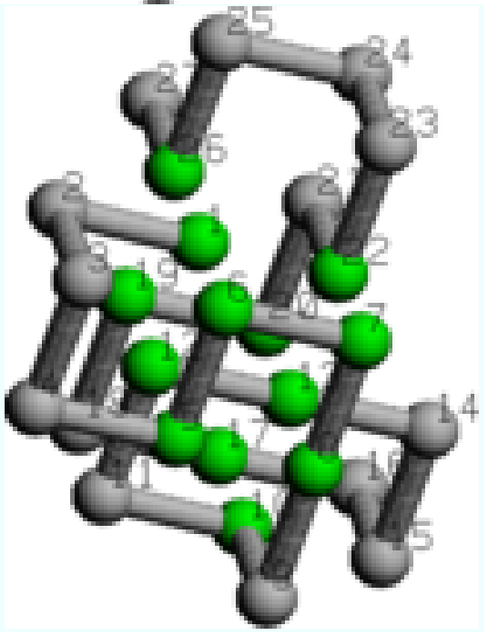
- example for reduction: sequence S2
  - HPNX: HXNNHHHHXHXHHNXNHXHHNHPPXHP
  - corresp. HP: HPPPHHHHPHPHHPPPHPHHPHPPPHP

# $S_2$ HP-sequence: 4 out of 297

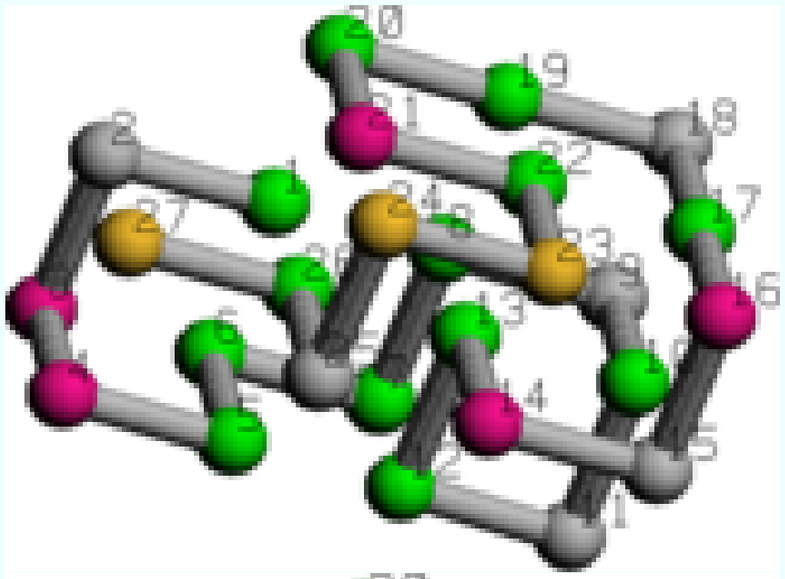


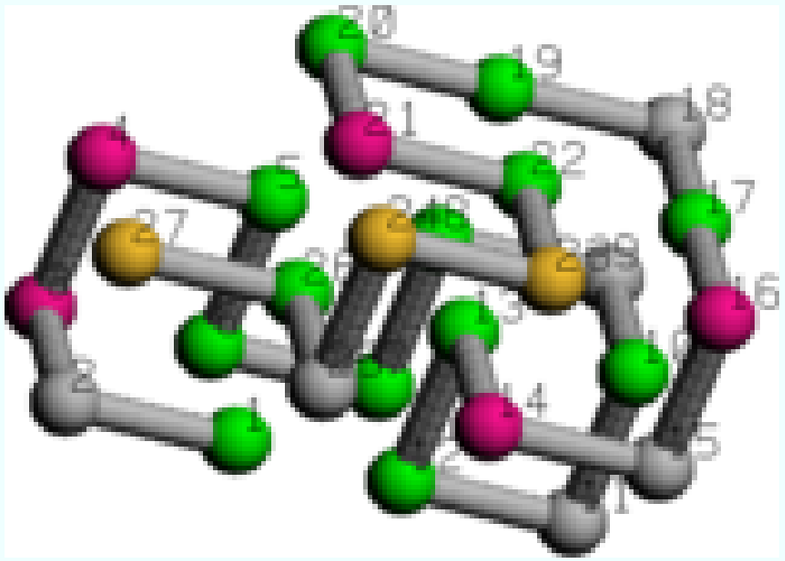


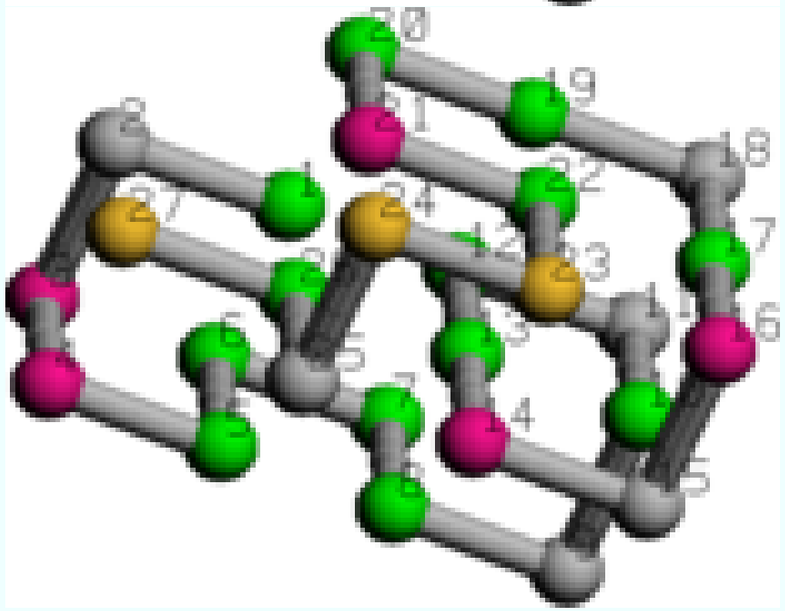


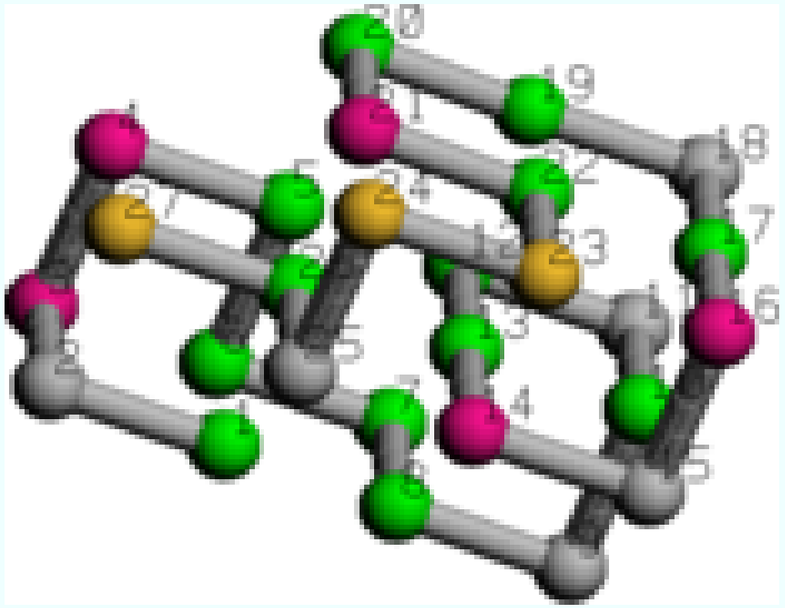


# $S_2$ HPNX-sequence: the 4 native ones

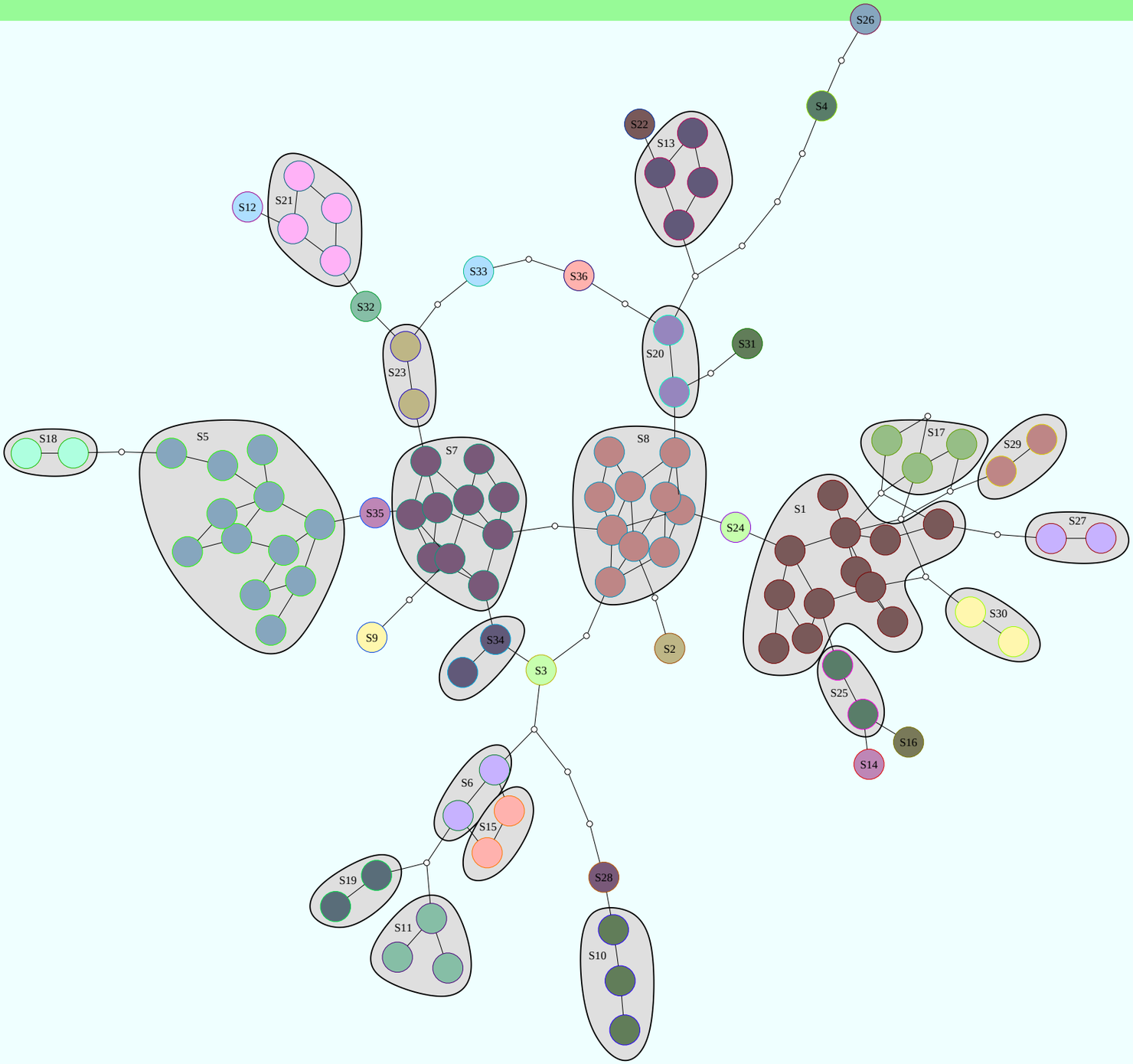








# Connectivity of Neutral Nets



# WWW-Page

University Jena · Computer Science
Bioinformatics

**Protein Structure Prediction In The FCC-HP-model**

PPPHPPPHHHPPHHPPPPPPHHHHPHPPHHPHPHHPHPHHHHHPHHPPPHPPPPHHPHHHPPHPPH HP-Sequence

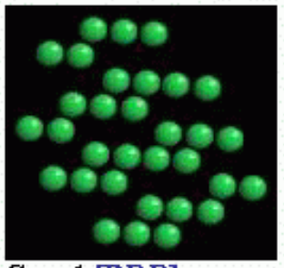
Reset | Fold It | Random

**Sequence**

PPPHPPPHHHPPHHPPPPPPHHHHPHPPHHPHPHHPHPHHHHHPHHPPPHPPPPHHPHHHPPHPPH

The submitted sequence has a length of **67**. The number of Hs in this sequence is **32**, which consequently is the size of the hydrophobic cores. Due to its number of Hs, any structure for this sequence has at most **115** HH-Contacts.

**Optimally compact Cores**

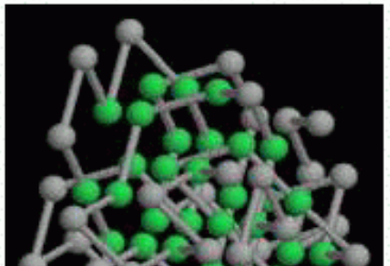


Core 1 [PDB]

The single core with 115 contacts.

Cores are precomputed.

**Optimal Structures**



# Conclusion

- constraint-based approach to protein folding
- guaranteed to find optima
- models:
  - HP-like models: HP, HPNX
  - lattices: cubic, FCC
- applications: properties of landscape
  - degeneracy
  - neutral nets
  - folding tunnel

# Acknowledgment


- Sebastian Will
- Erich Bornberg-Bauer
- Peter Stadler
- Michael Wolfinger