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

Rolf Backofen and Sebastian Will
Chair for Bioinformatics
Albert-Ludiwgs-Universität Freiburg

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

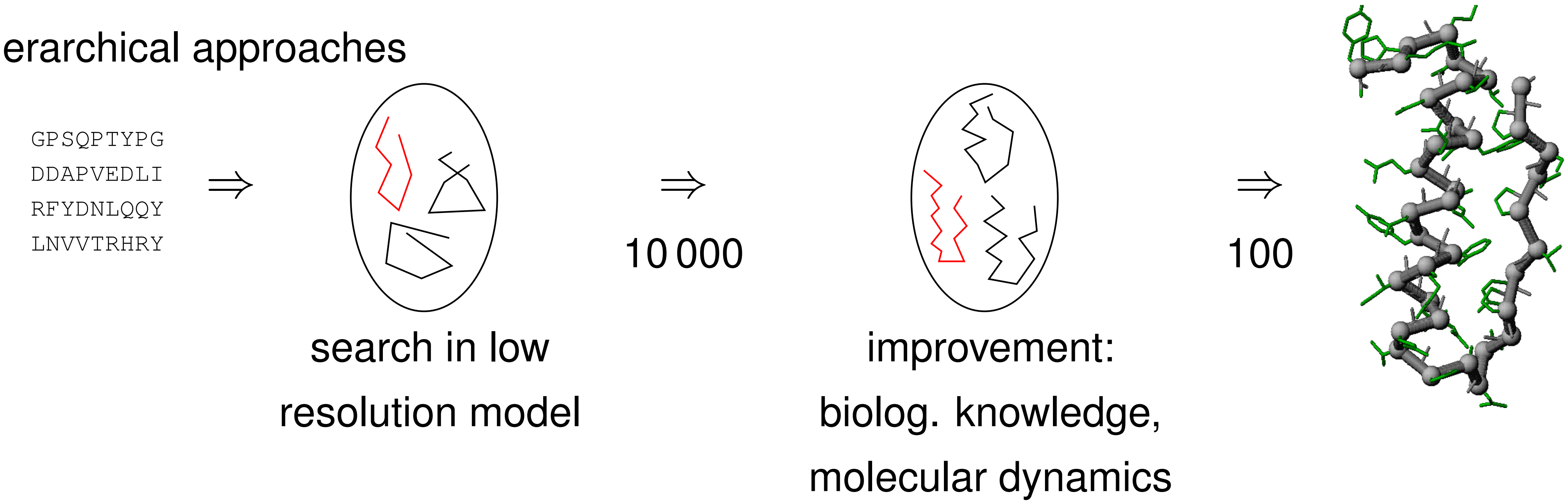
Structure Prediction as Optimization Problem

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

  ⇒ huge search space

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

- hierarchical approaches



GPSQPTYPG
DDAPVEDLI
RFYDNLQQY
LNVVTRHRY

⇒

search in low
resolution model

⇒
10 000

improvement:
biolog. knowledge,
molecular dynamics

⇒
100

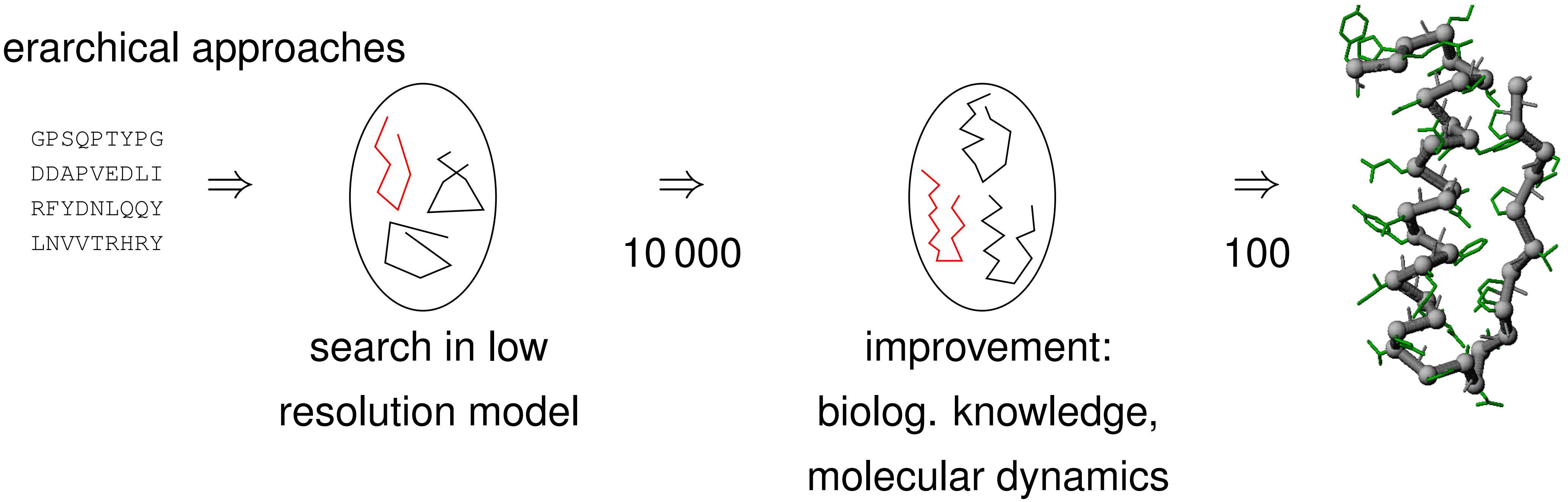
Structure Prediction as Optimization Problem

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

  ⇒ huge search space

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

- hierarchical approaches



GPSQPTYPG
DDAPVEDLI
RFYDNLQQY
LNVVTRHRY

⇒ search in low resolution model ⇒ 10 000 improvement: biolog. knowledge, molecular dynamics ⇒ 100

- often: low-resolution model = lattice model

Previous Prediction Approaches for Latttice Models

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

  advantages:
  - $\ast$ fast

  disadvantages:
  - $\ast$ only for structure prediction

- mostly: monte-carlo/simulated annealing

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

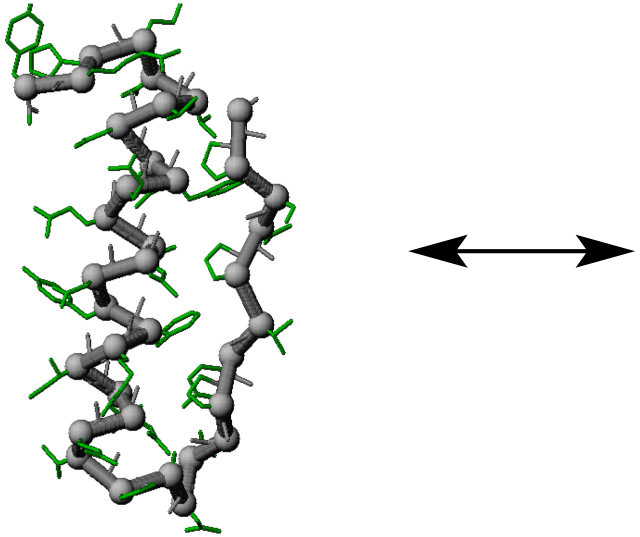
  disadvantages:
  - $\ast$ for HP-model, optimal solution nearly never found
  - $\ast$ most approaches are **not** ergodic

- also: complete enumeration

  advantages:
  - $\ast$ direct exploration of landscape

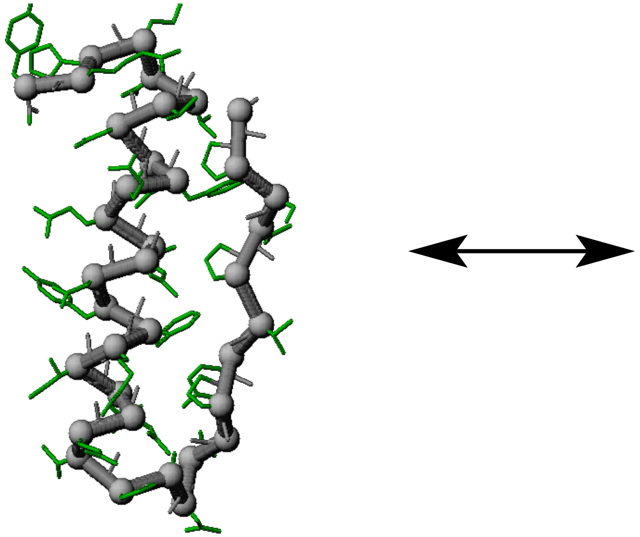
  disadvantages:
  - $\ast$ very short sequences, only 2D

Idea of Lattice Models



- trade-off: choose between
    - models, that **closely** resembles proteins structure
      BUT no hope of ever finding the native structure

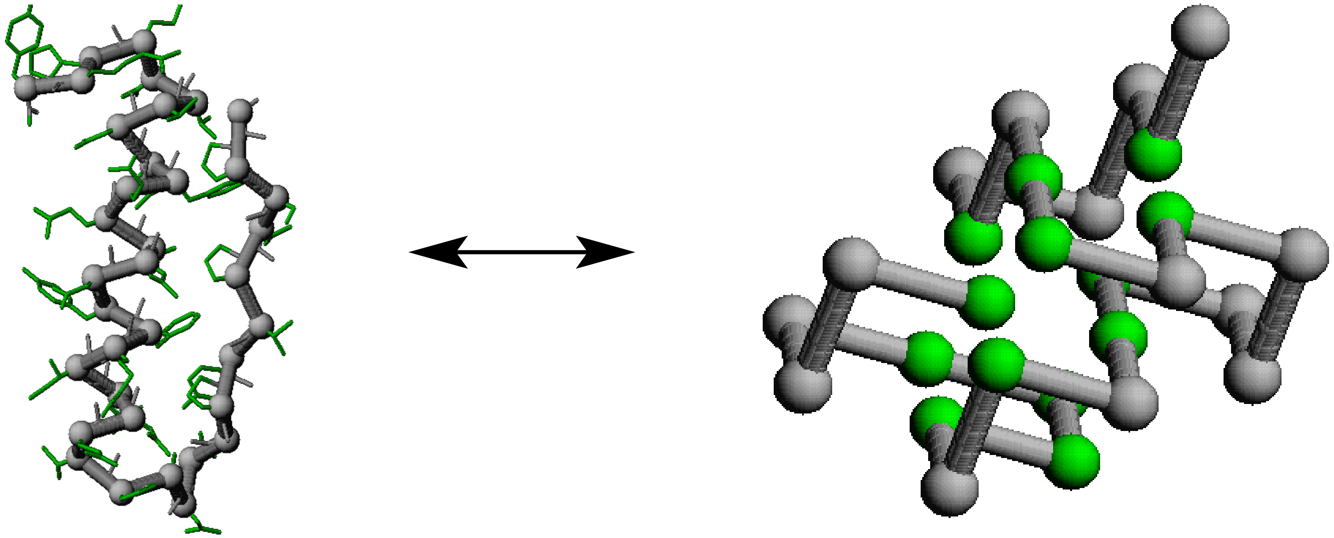
Idea of Lattice Models



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

      BUT no hope of (algorithmically) finding the native structure

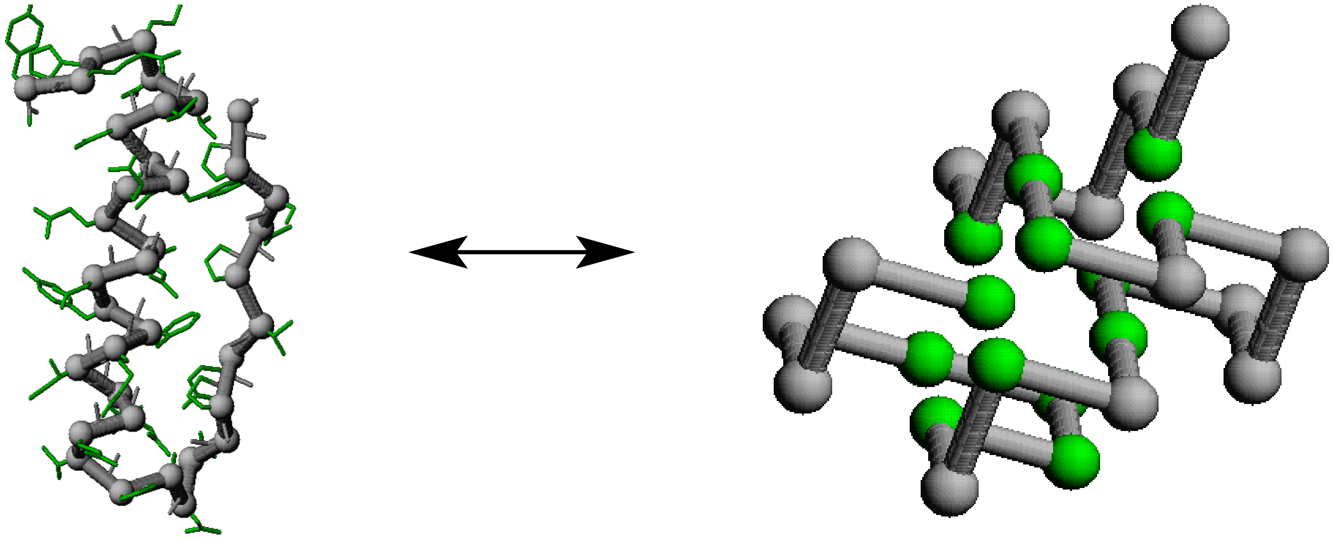
Idea of Lattice Models



- trade-off: choose between
  - models, that **closely** resembles proteins structure
    BUT no hope of (algorithmically) finding the native structure

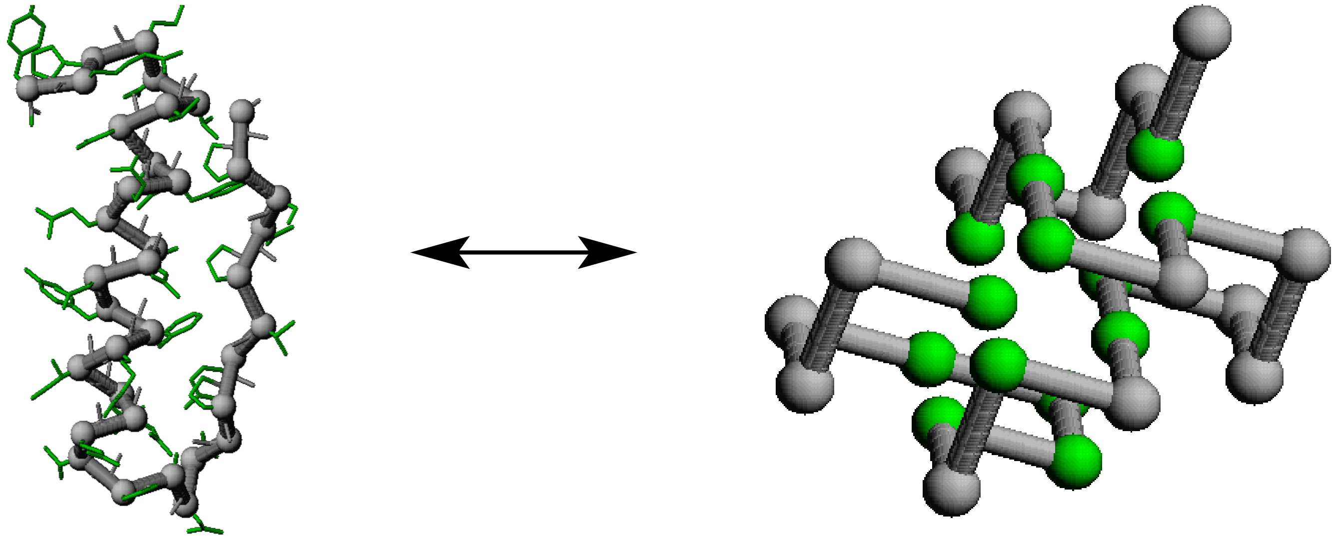
  - models, that **crudely** resembles proteins structure

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

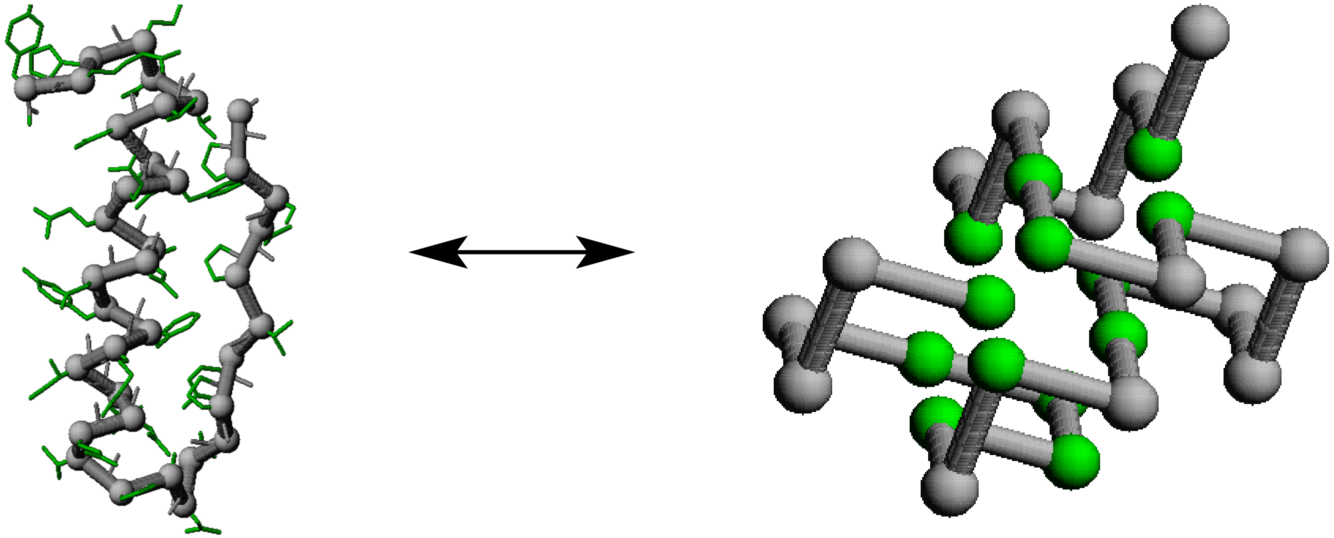
    BUT no hope of (algorithmically) finding the native structure

  - 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~~

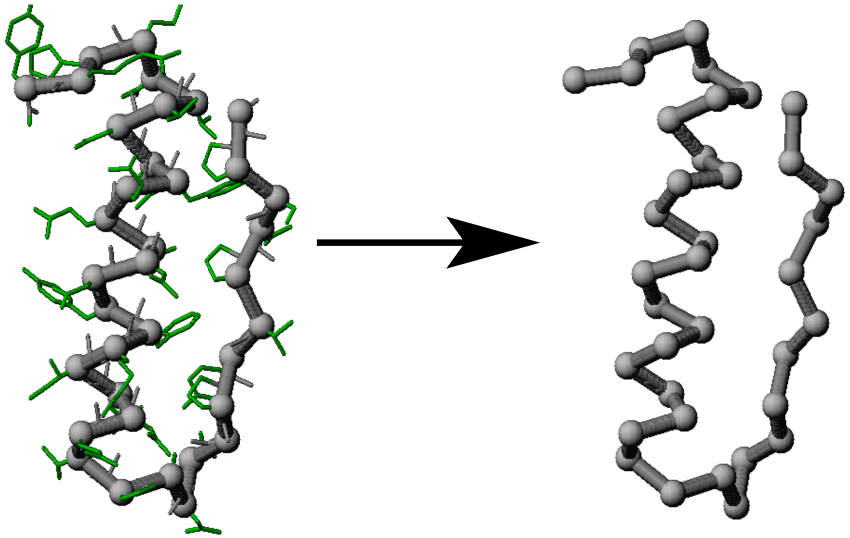
    BUT SO FAR: we cannot find the native structure either

- **here:** **BUT** we can find the native structure

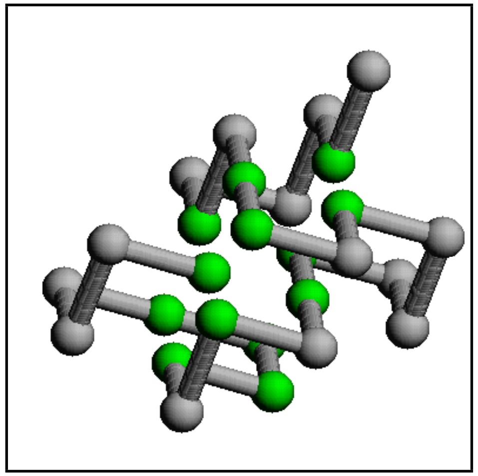
  using constraint programming

Lattice Models

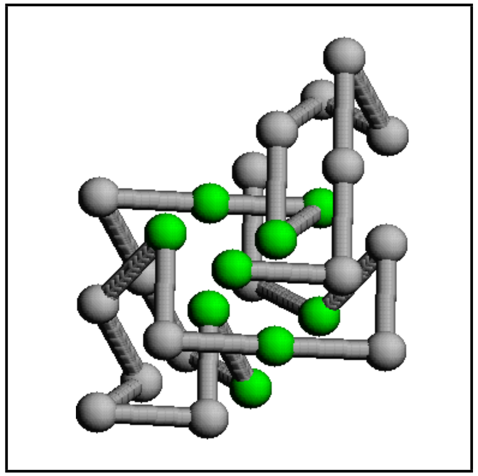
- lattice models:
  - usually only backbone
  - positions = positions on lattice
  - self-avoiding: no steric conflicts



- often used lattices:



cubic

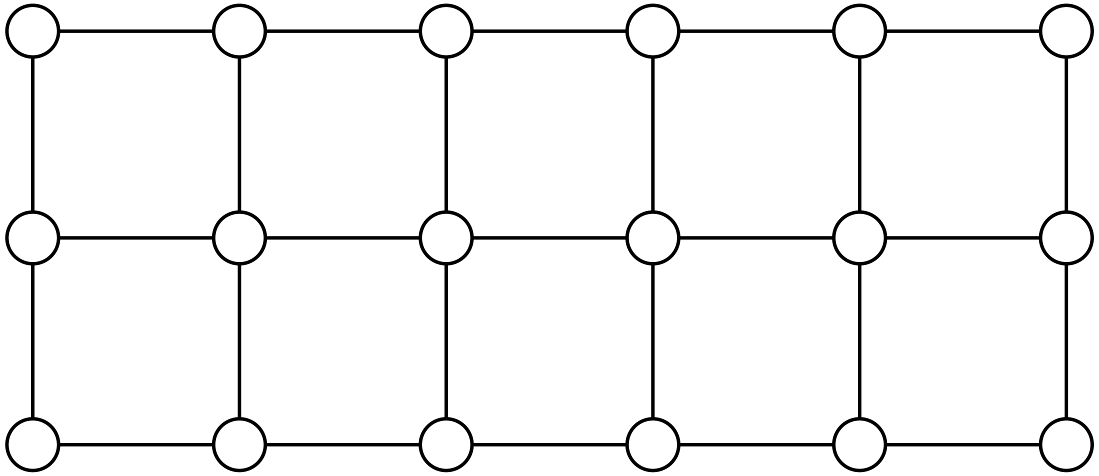


face-centered-cubic

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

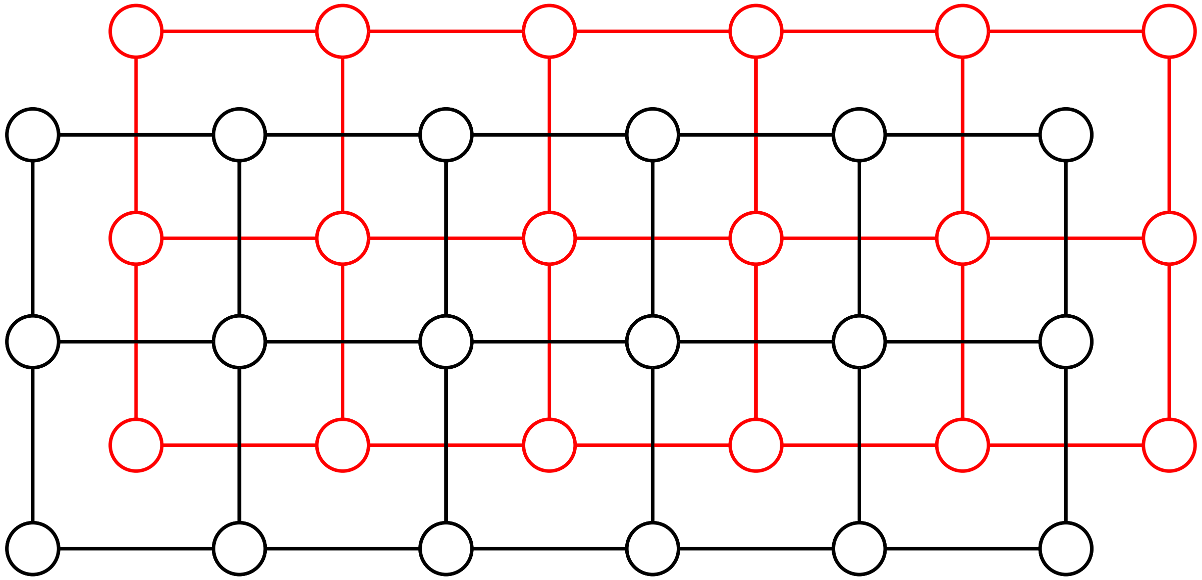
The FCC

- FCC = face-centered cubic lattice



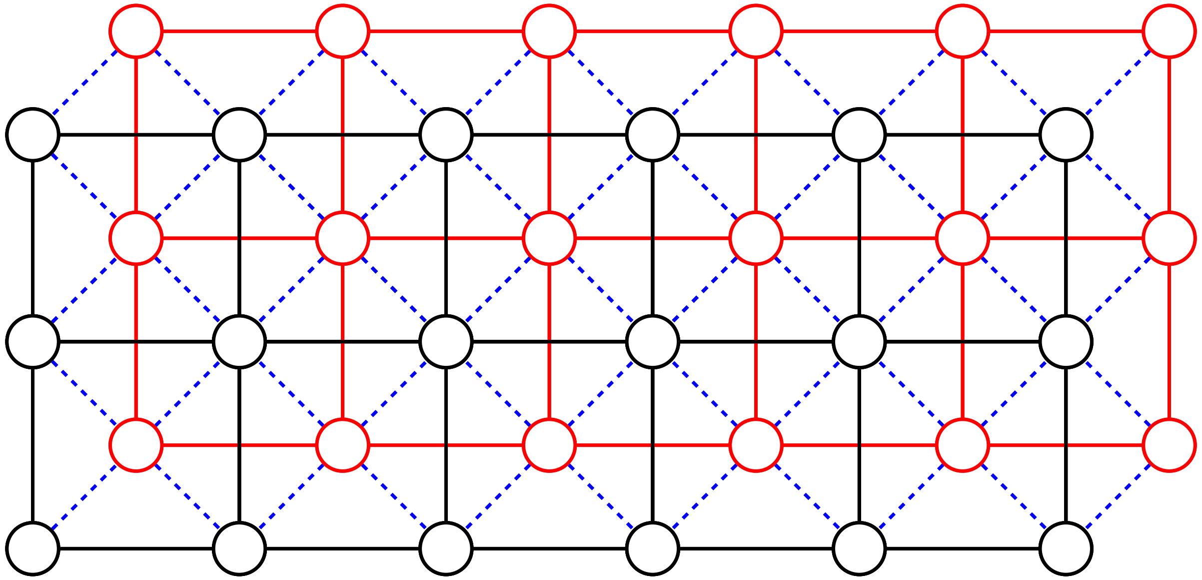
The FCC

- FCC = face-centered cubic lattice



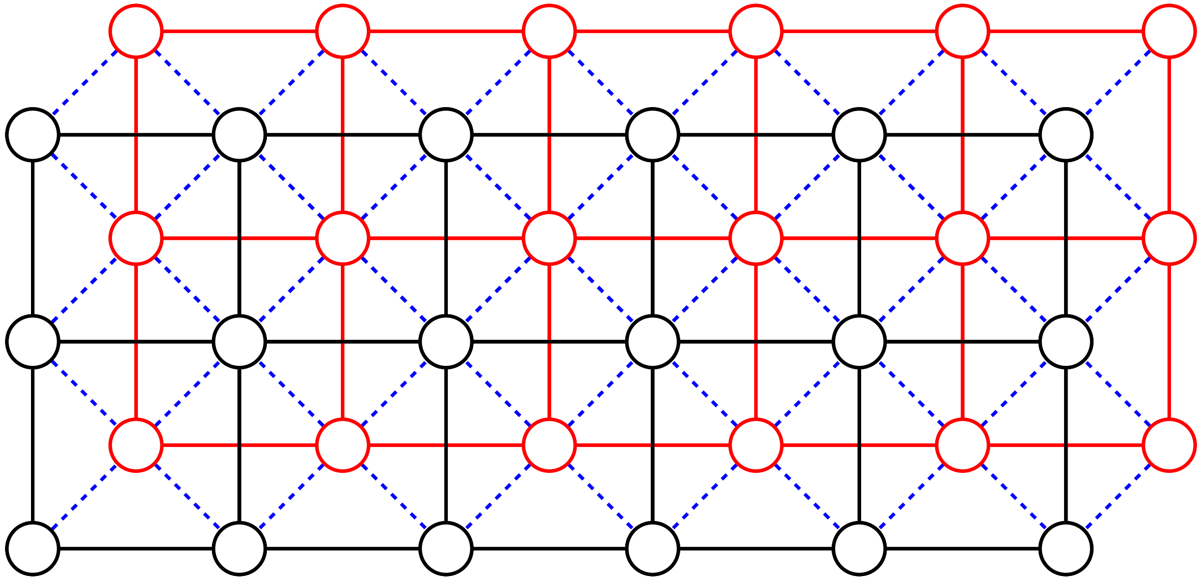
The FCC

- FCC = face-centered cubic lattice



The FCC

- FCC = face-centered cubic lattice

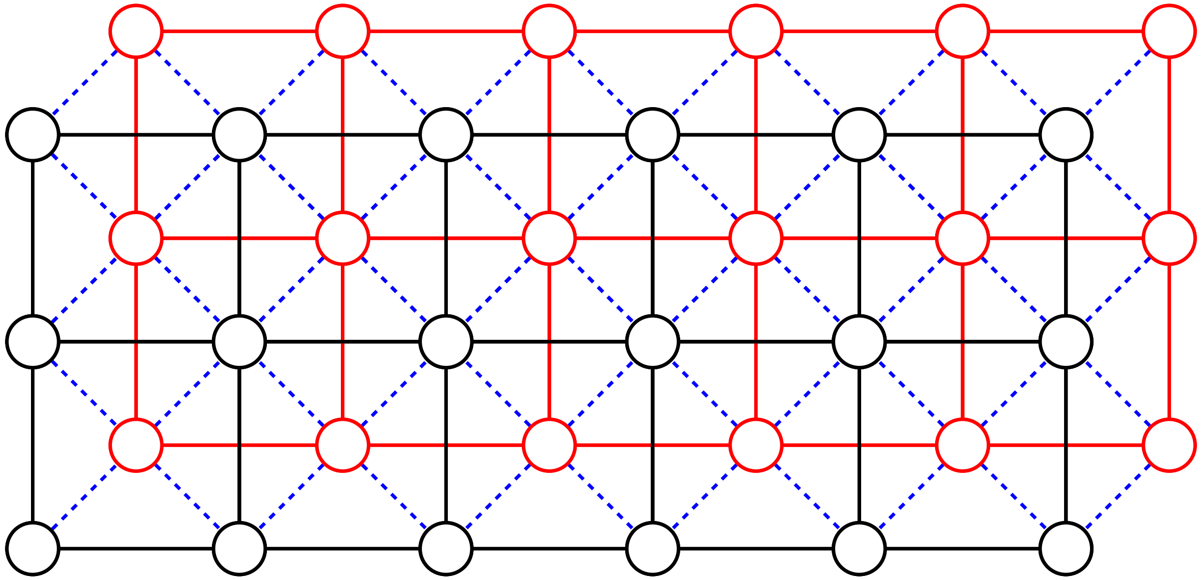


- Kepler's conjecture: FCC=densest packing of balls

  proved just recently (after ≈ 400 years)

The FCC

- FCC = face-centered cubic lattice



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

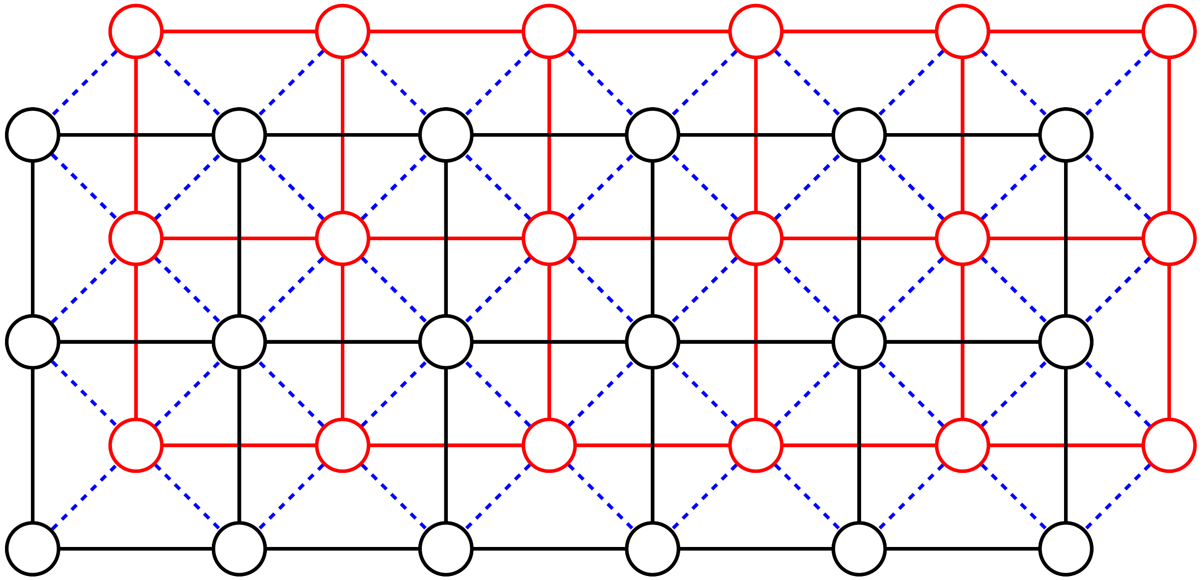
  proved just recently (after ≈ 400 years)

- [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.…

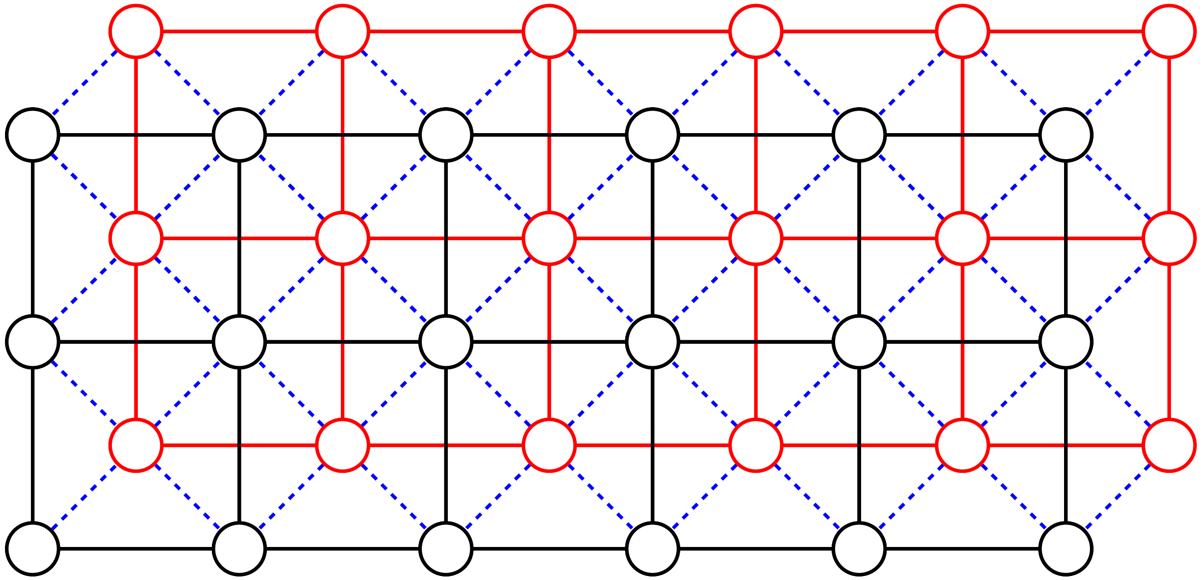
  …We confirm that lattices with large coordination numbers provide better fits to protein structures …

Relation to Proteins



- how does it related to proteins?

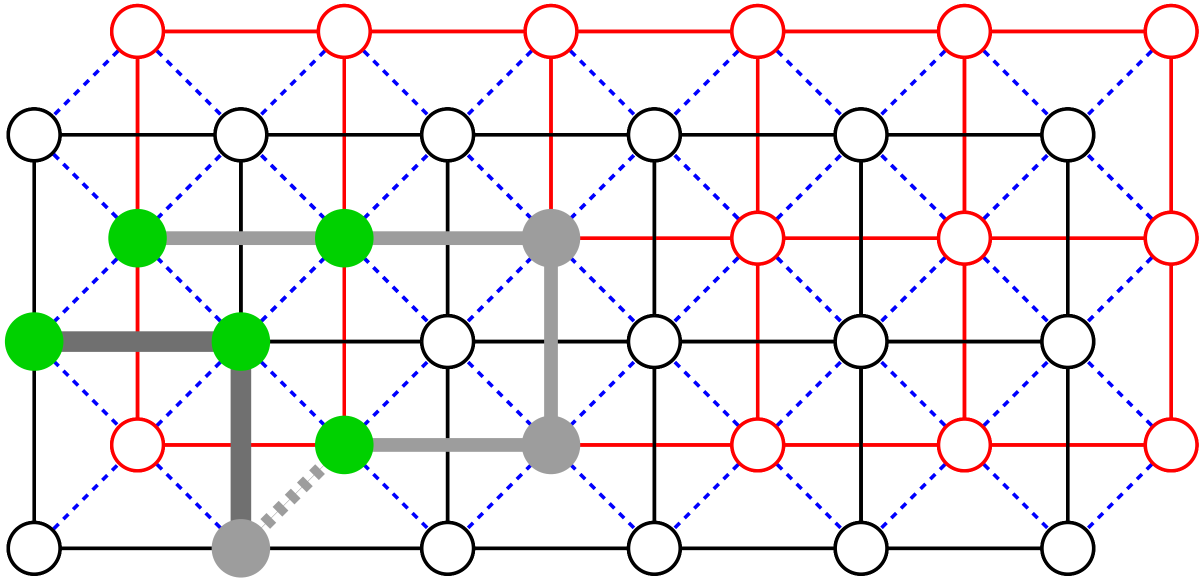
Relation to Proteins



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

alphabet:
H = Hydrophobic ●
P = Polar (hydrophilic) ●

Relation to Proteins

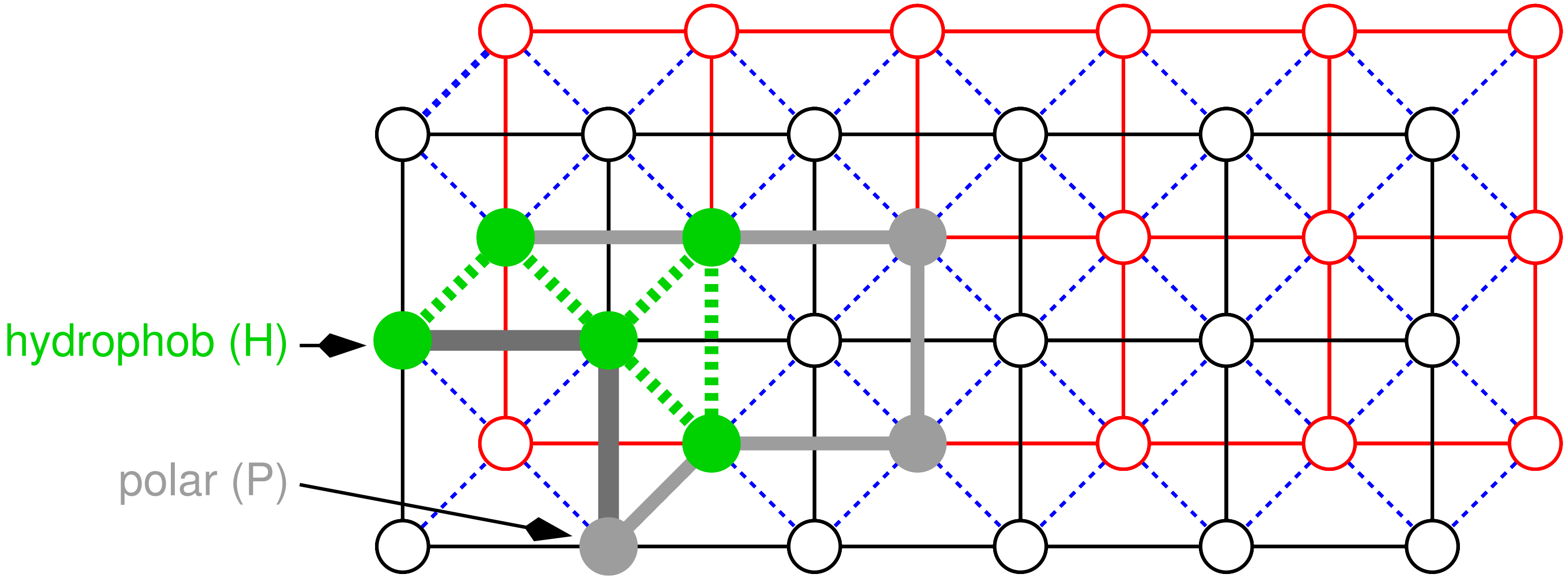


- how does it related to proteins?
  - hydrophobic and polar (hydrophilic) amino acids
  - hydrophobic are densely packed

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

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

Relation to Proteins

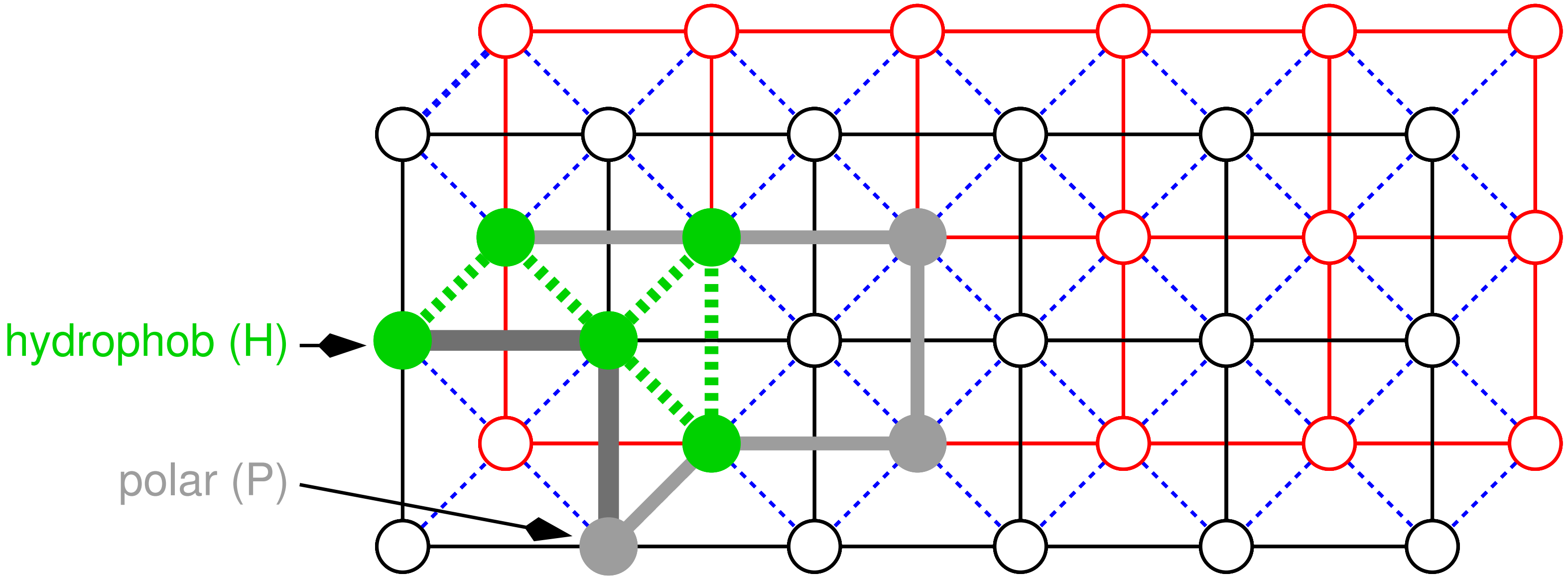


- how does it related to proteins?
  - hydrophobic and polar (hydrophilic) amino acids
  - hydrophobic are densely packed

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

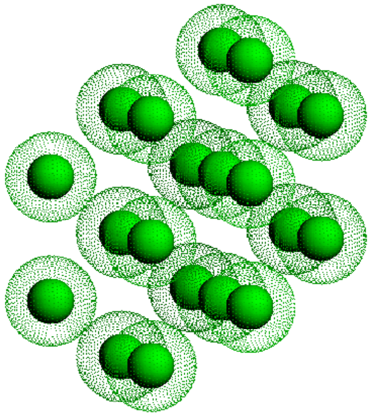
General Approach

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

Step 1: compute lower energy bounds

estimate contacts (within layers, between layers)

⇒
Step 1



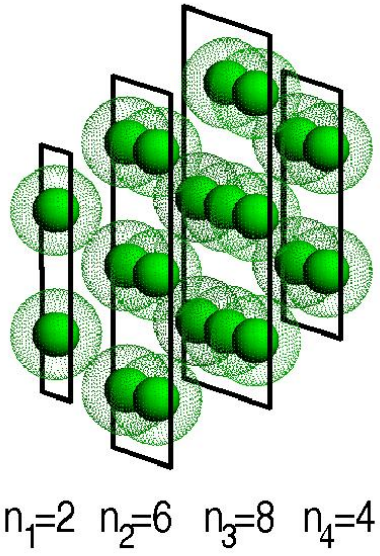
General Approach

- 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



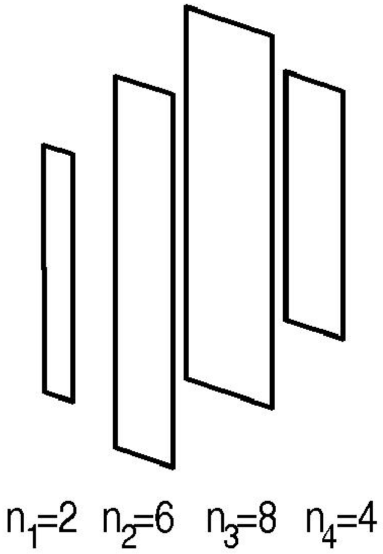
General Approach

- 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

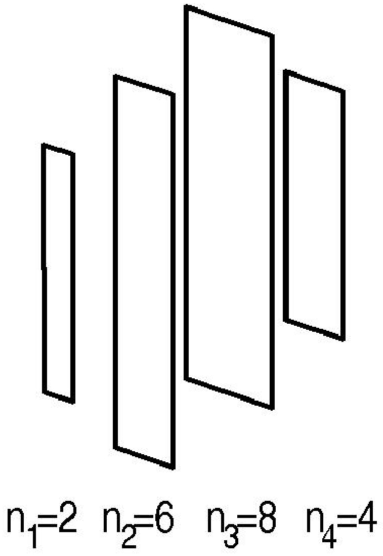
Step 1: compute lower energy bounds

estimate contacts (within layers, between layers)

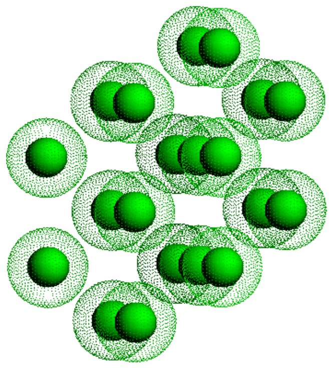
Step 2: construct hydrophobic cores

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)

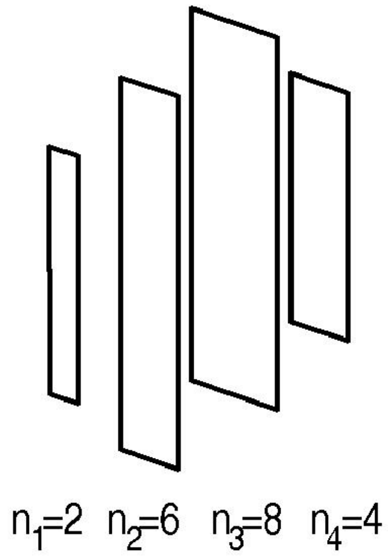
Step 2: construct hydrophobic cores

use bounds from last step, precomputed

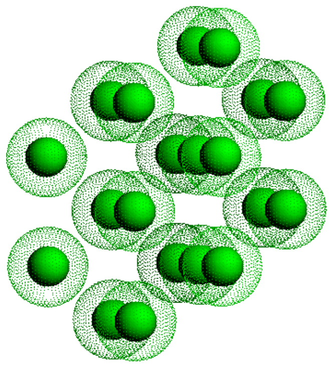
Step 3: thread sequence to hydrophobic cores of size n.

using constraint propagation

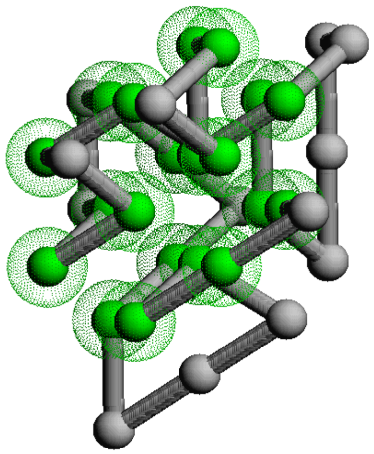
⇒ Step 1



⇒ Step 2



⇒ Step 3

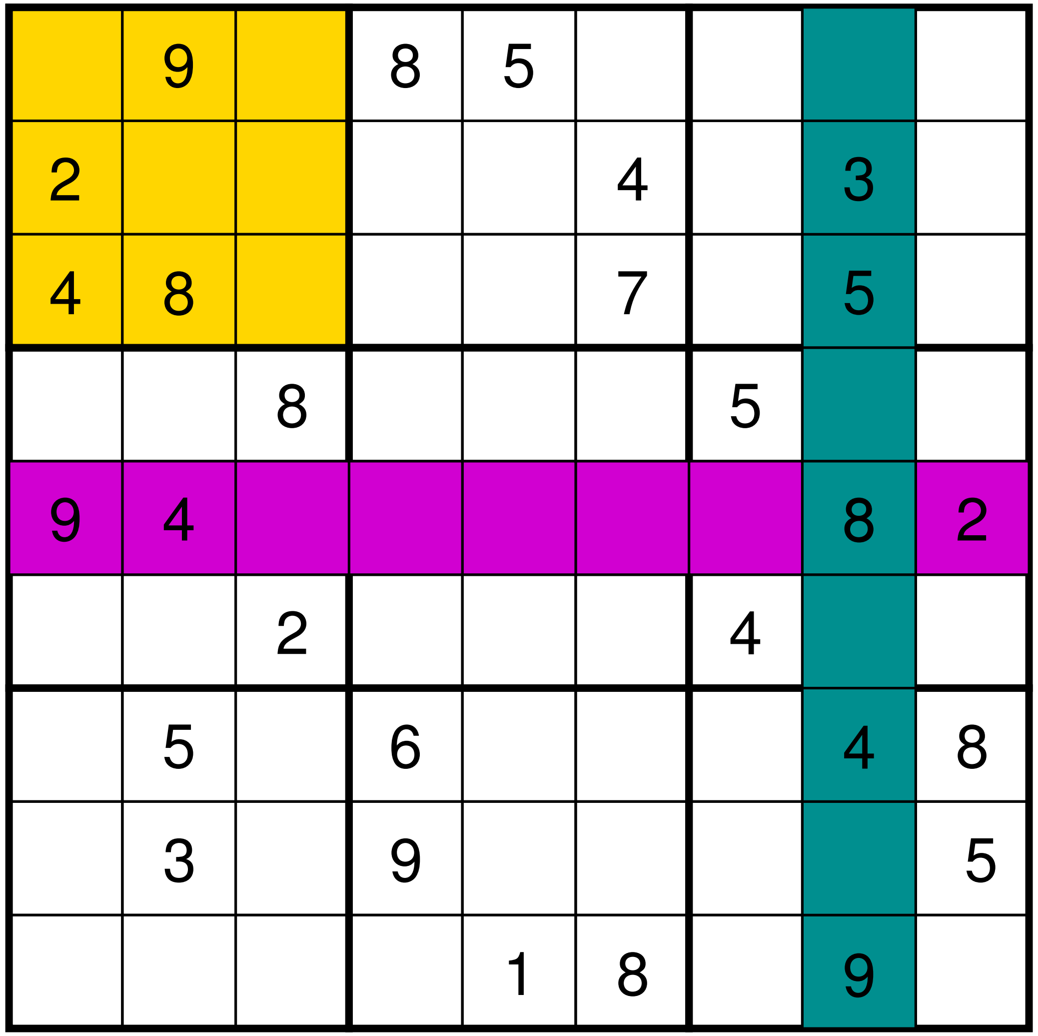


Example for a Constraint-Problem: Sudoku

	9		8	5				
2					4		3	
4	8				7		5	
		8				5		
9	4						8	2
		2				4		
	5		6				4	8
	3		9					5
				1	8		9	

- every number from $1 \ldots 9$ exactly once in

Example for a Constraint-Problem: Sudoku

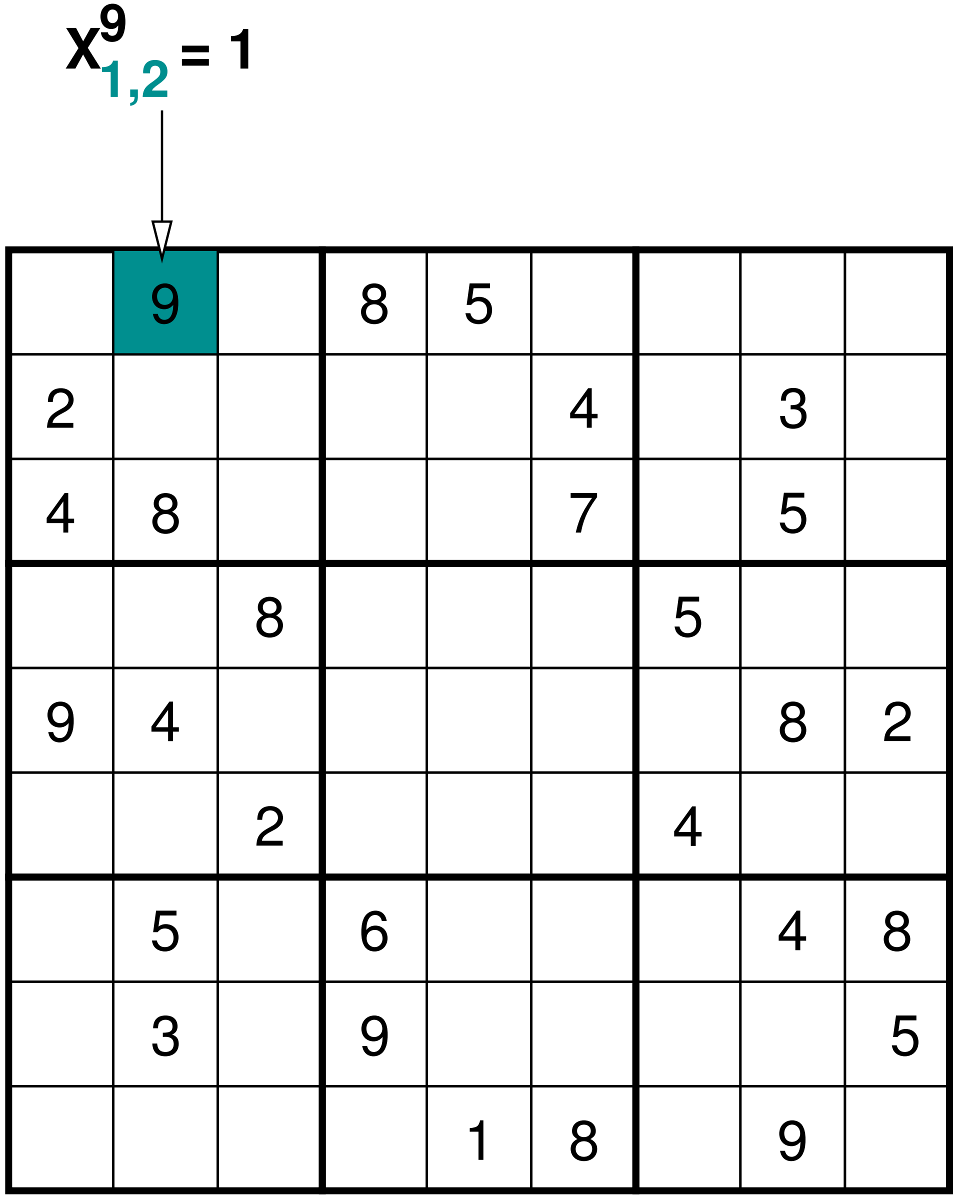


	9		8	5				
2					4		3	
4	8				7		5	
		8				5		
9	4						8	2
		2				4		
	5		6				4	8
	3		9					5
				1	8		9	

- every number from $1 \ldots 9$ exactly once in
    - every row
    - every column
    - every block

Constraints-Formulation

- Variablen $X^n_{k,l} \in \{0,1\}$ for every row k, column l and number $n \in \{1 \ldots 9\}$.
- Constraints

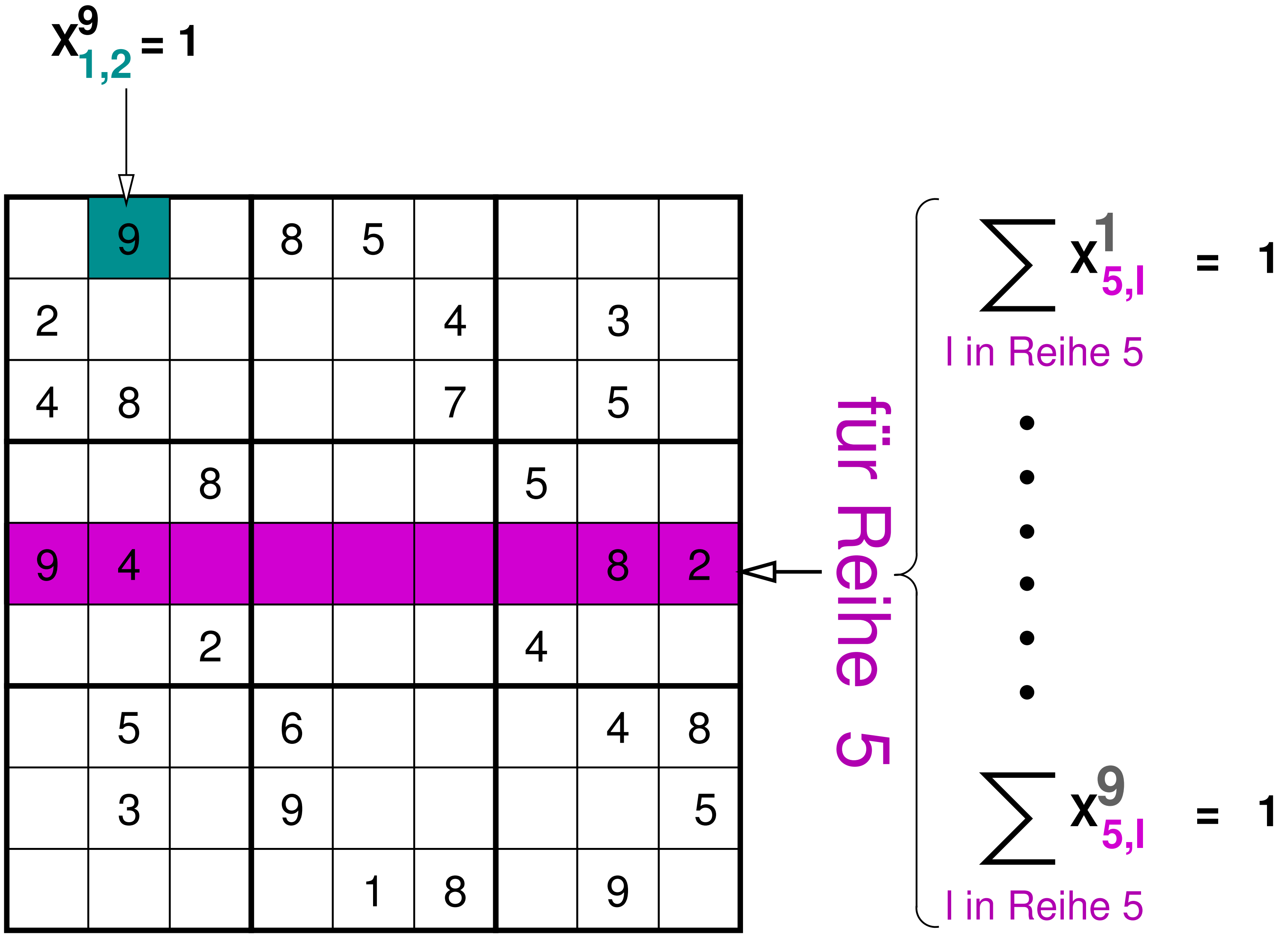


$X^9_{1,2} = 1$

	9		8	5				
2					4		3	
4	8				7		5	
		8				5		
9	4						8	2
		2				4		
	5		6				4	8
	3		9					5
				1	8		9	

Constraints-Formulation

- Variablen $X^n_{k,l} \in \{0,1\}$ for every row k, column l and number $n \in \{1 \ldots 9\}$.
- Constraints



$\mathbf{x}^{\mathbf{9}}_{\mathbf{1,2}} = \mathbf{1}$

	9		8	5				
2					4		3	
4	8				7		5	
		8				5		
9	4						8	2
		2				4		
	5		6				4	8
	3		9					5
				1	8		9	

für Reihe 5

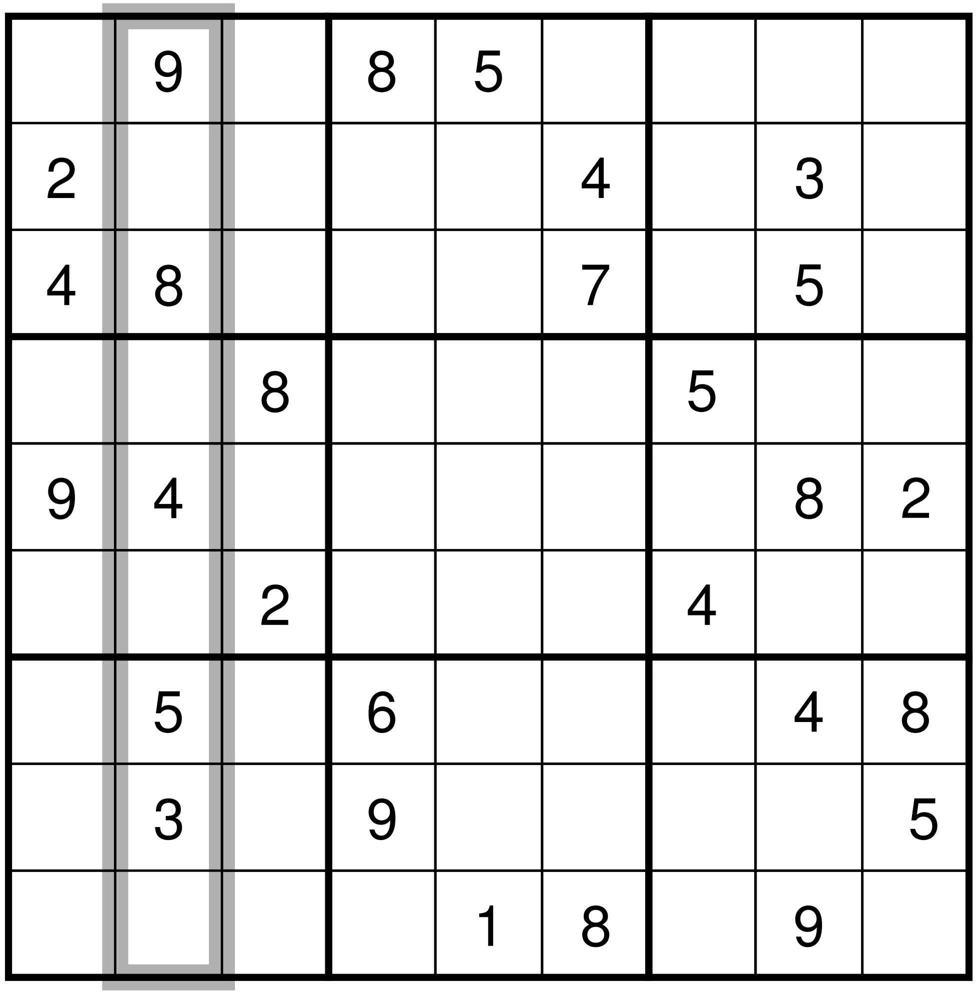
$$\sum_{\text{l in Reihe 5}} \mathbf{x}^{1}_{5,l} = 1$$

$$\vdots$$

$$\sum_{\text{l in Reihe 5}} \mathbf{x}^{9}_{5,l} = 1$$

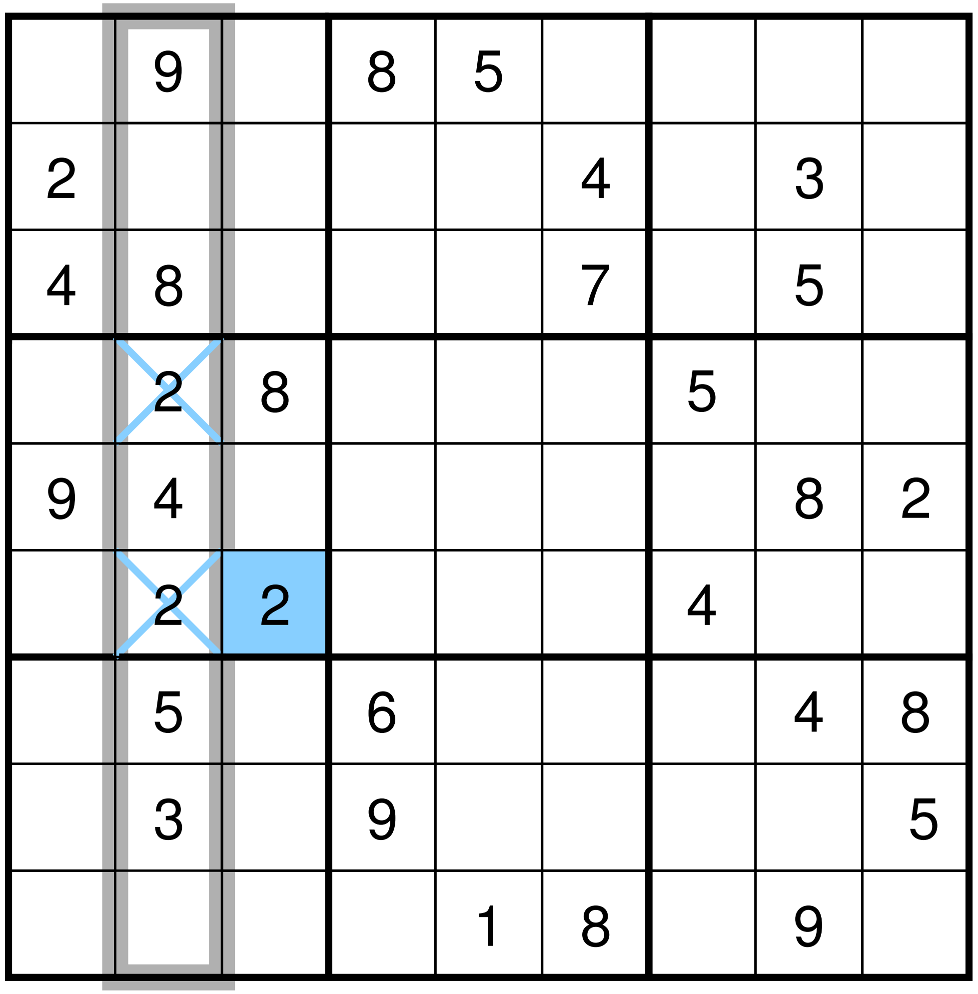
- similar for all columns and blocks

Constraint-Propagation



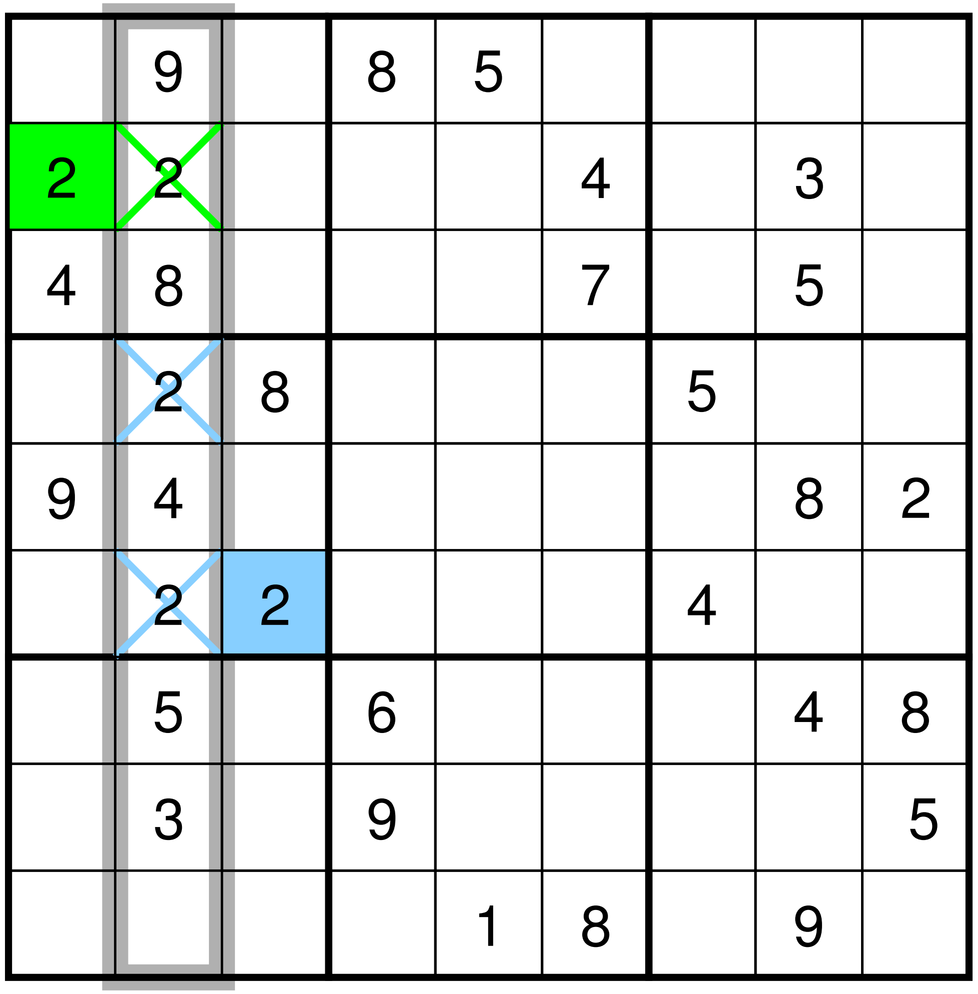
	9		8	5				
2					4		3	
4	8				7		5	
		8				5		
9	4						8	2
		2				4		
	5		6				4	8
	3		9					5
				1	8		9	

Constraint-Propagation



	9		8	5				
2					4		3	
4	8				7		5	
	~~2~~	8				5		
9	4						8	2
	~~2~~	2				4		
	5		6				4	8
	3		9					5
				1	8		9	

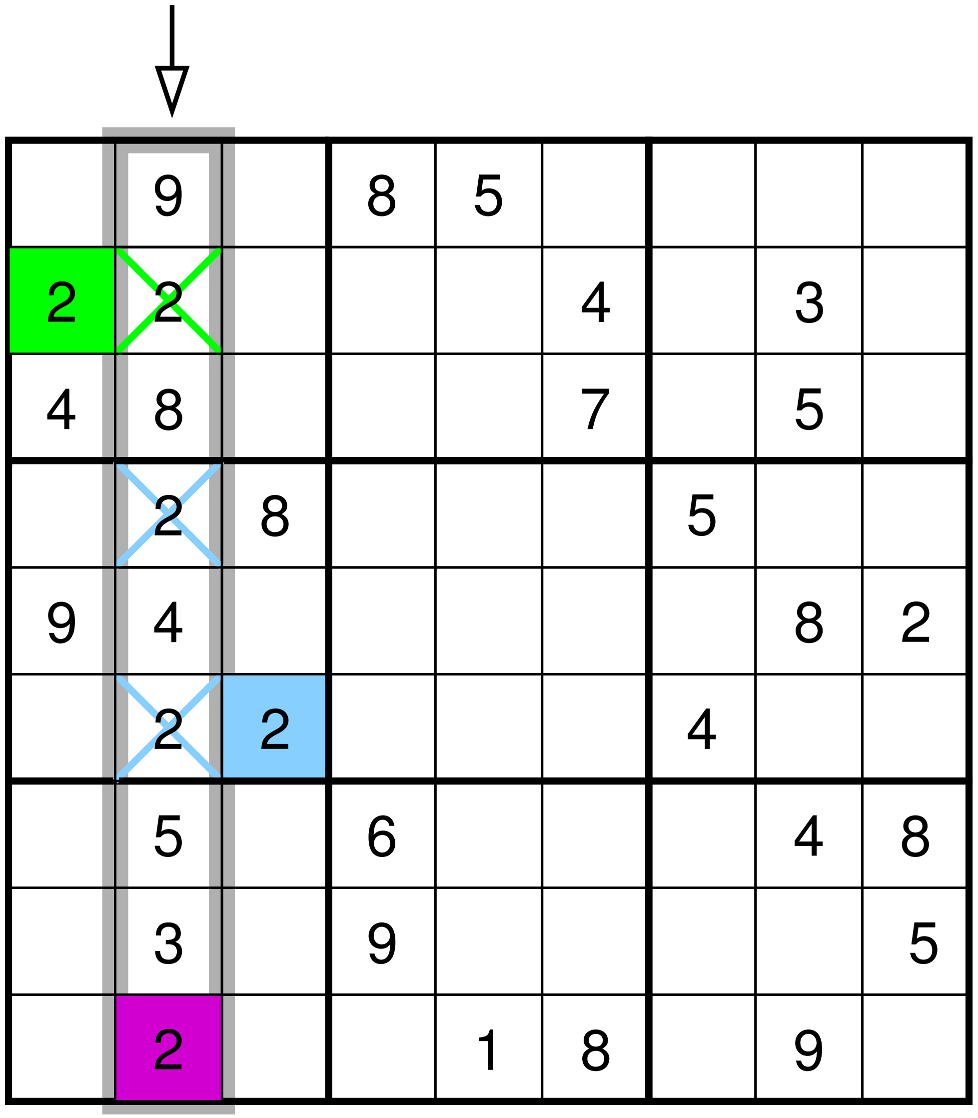
Constraint-Propagation



	9		8	5				
2	~~2~~				4		3	
4	8				7		5	
	~~2~~	8				5		
9	4						8	2
	~~2~~	2				4		
	5		6				4	8
	3		9					5
				1	8		9	

Constraint-Propagation

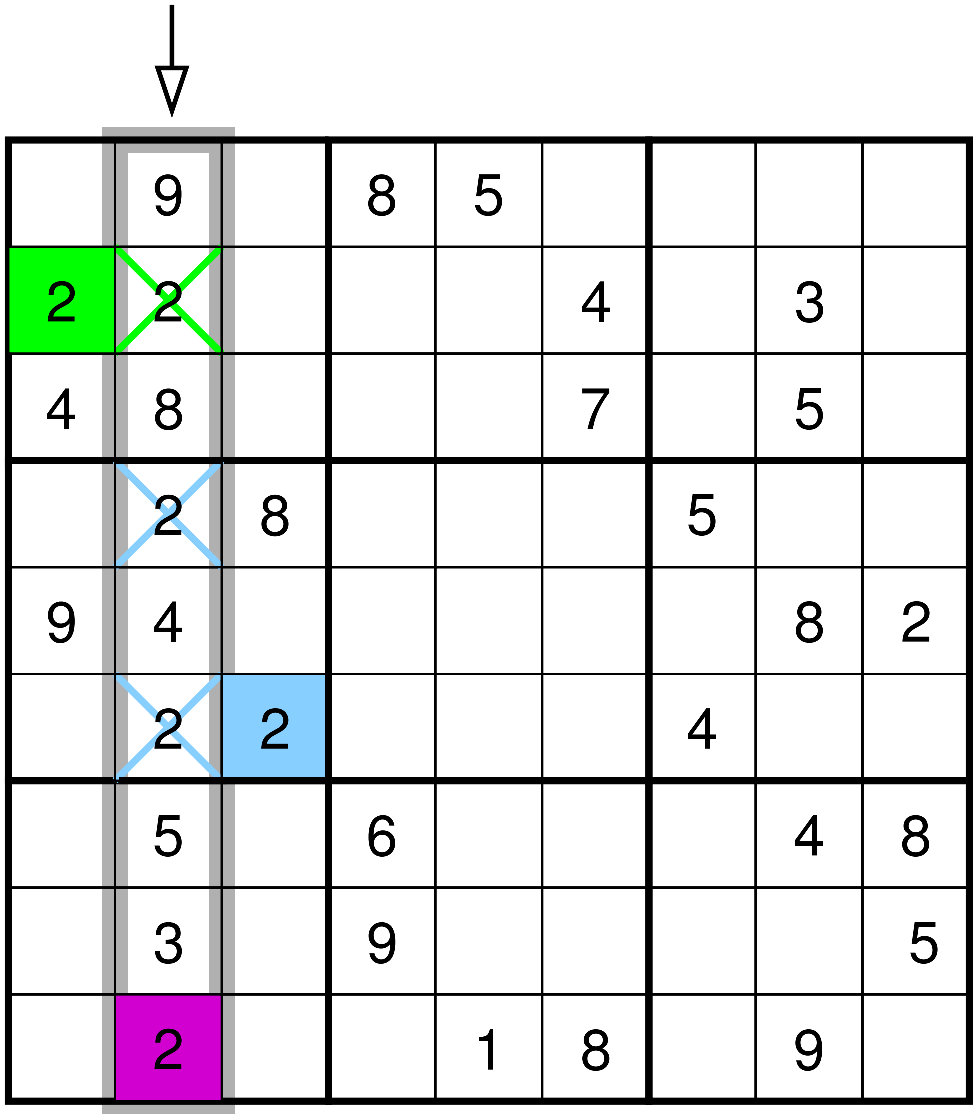
$$\sum_{k = 1..9} x^{2}_{k,2} = 1$$



	9		8	5				
2	~~2~~				4		3	
4	8				7		5	
	~~2~~	8				5		
9	4						8	2
	~~2~~	2				4		
	5		6				4	8
	3		9					5
	2			1	8		9	

Constraint-Propagation

$$\sum_{k = 1..9} x_{k,2}^{2} = 1$$



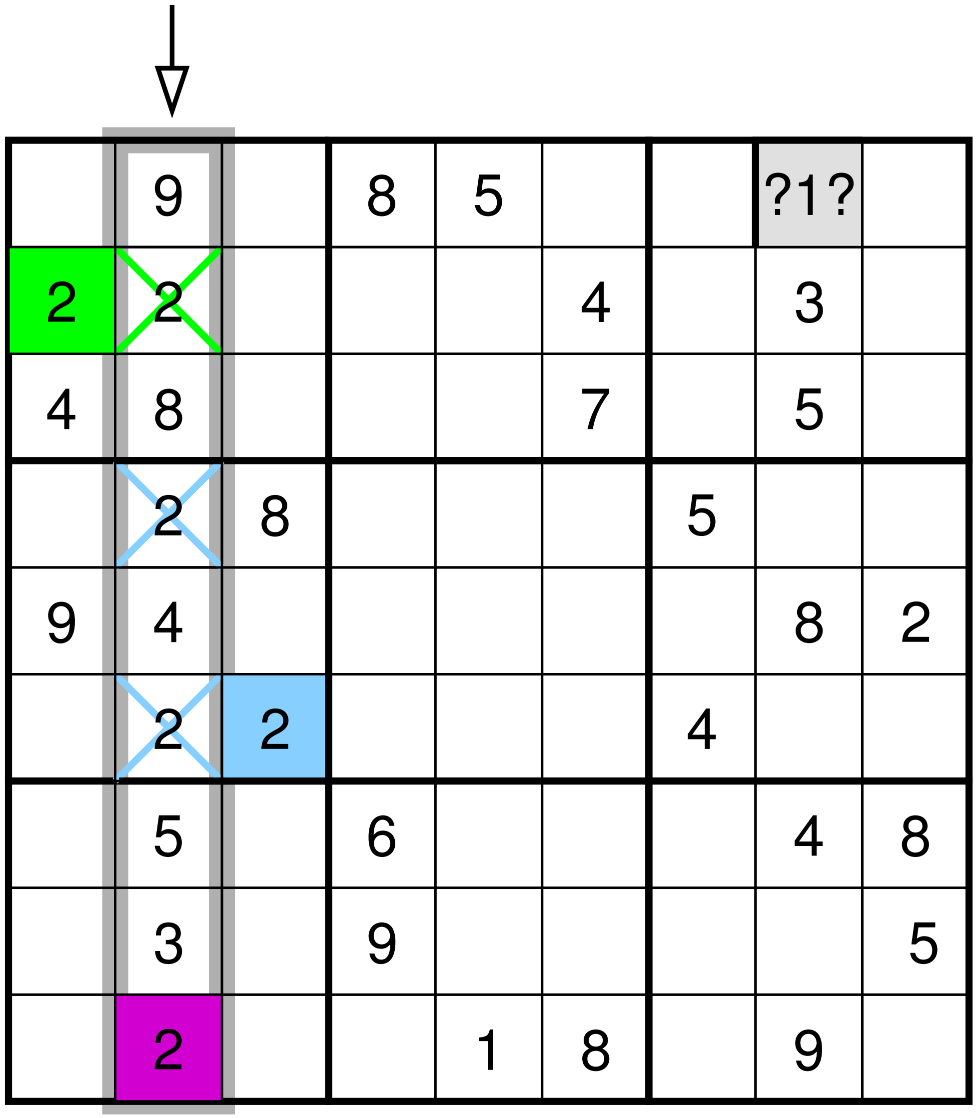
	9		8	5				
2	~~2~~				4		3	
4	8				7		5	
	~~2~~	8				5		
9	4						8	2
	~~2~~	2				4		
	5		6				4	8
	3		9					5
	2			1	8		9	

- new values are provable correct
- iteration till stable state

- if not solved: **Search**
  - for one variable: split over all possible values

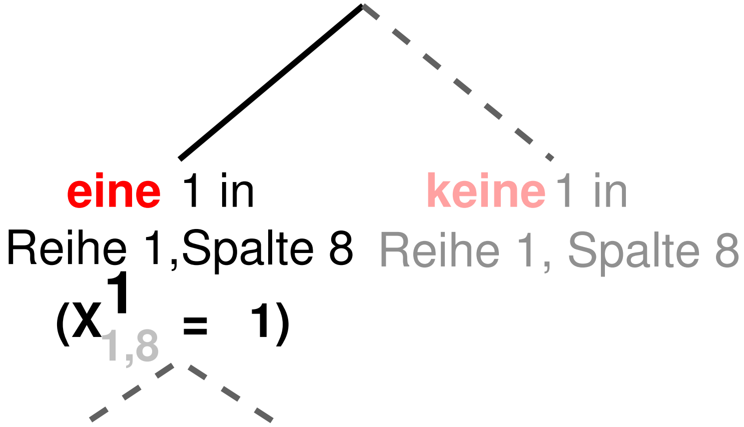
Constraint-Propagation

$$\sum_{k = 1..9} x^{2}_{k,2} = 1$$



	9		8	5			?1?	
2	~~2~~				4		3	
4	8				7		5	
	~~2~~	8				5		
9	4						8	2
	~~2~~	2				4		
	5		6				4	8
	3		9					5
	2			1	8		9	

- new values are provable correct
- iteration till stable state
- if not solved: **Search**
  - for one variable: split over all possible values
  - followed by Propagation.



eine 1 in Reihe 1,Spalte 8 ($x^{1}_{1,8}$ = 1)

keine 1 in Reihe 1, Spalte 8

- “naive” search (generate-and-test):

  9^{53} steps

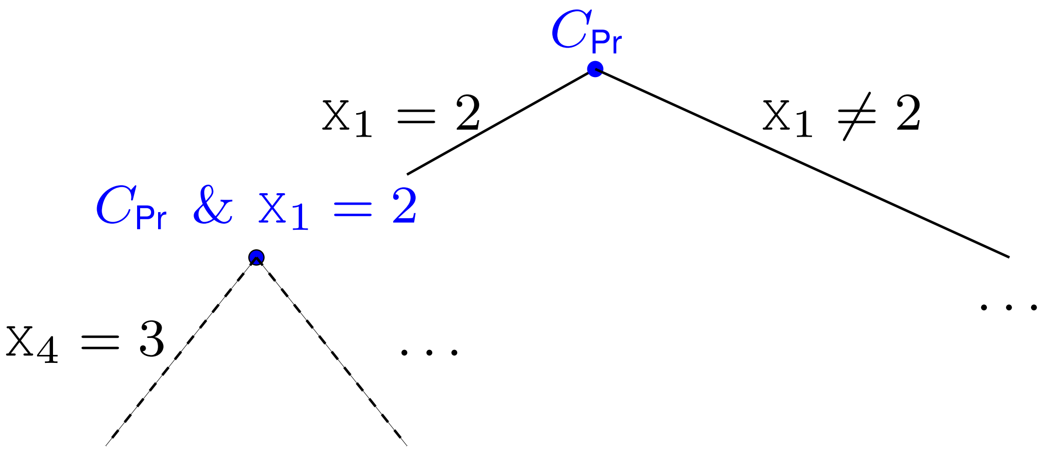
Constraint-Programming: automatisation of progagation and search

Constraint-based Formulation

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

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

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

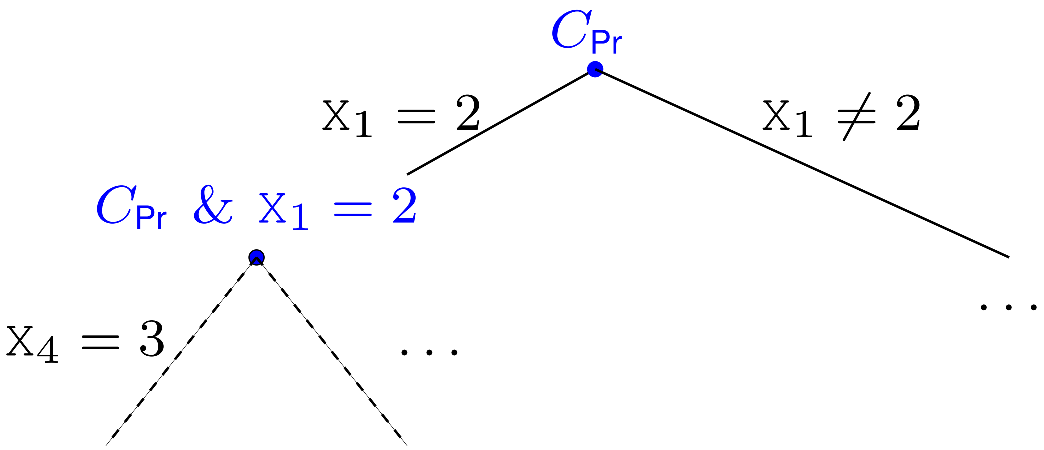


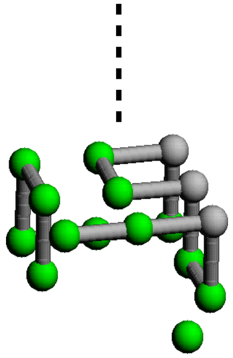
Constraint-based Formulation

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

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

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



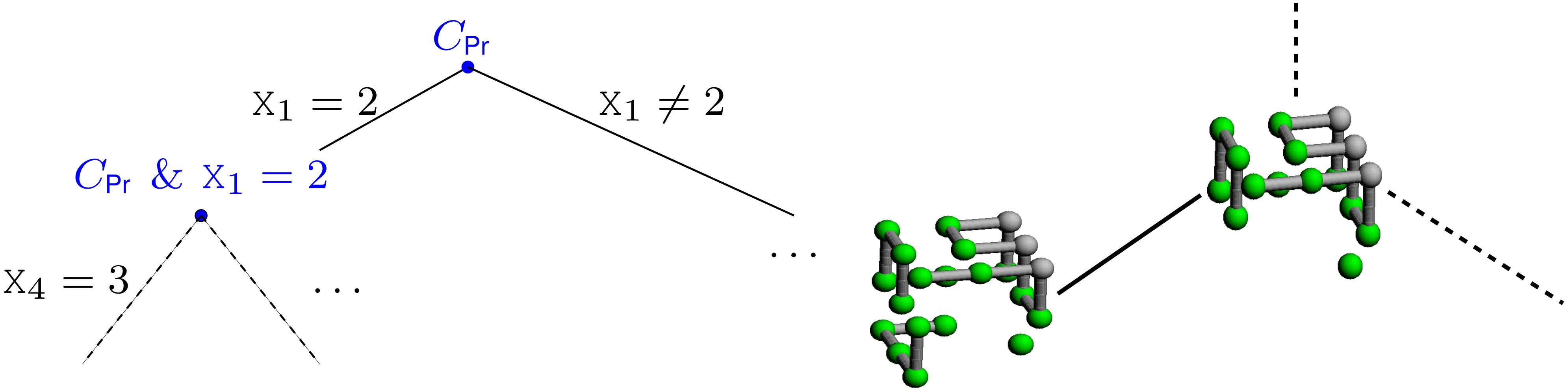


Constraint-based Formulation

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

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

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

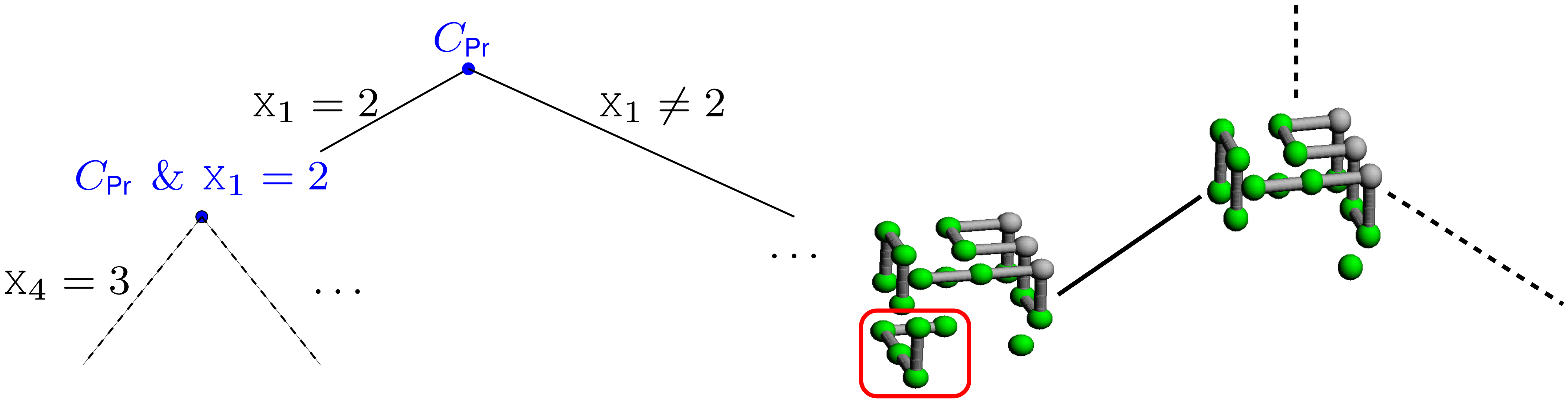


Constraint-based Formulation

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

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

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

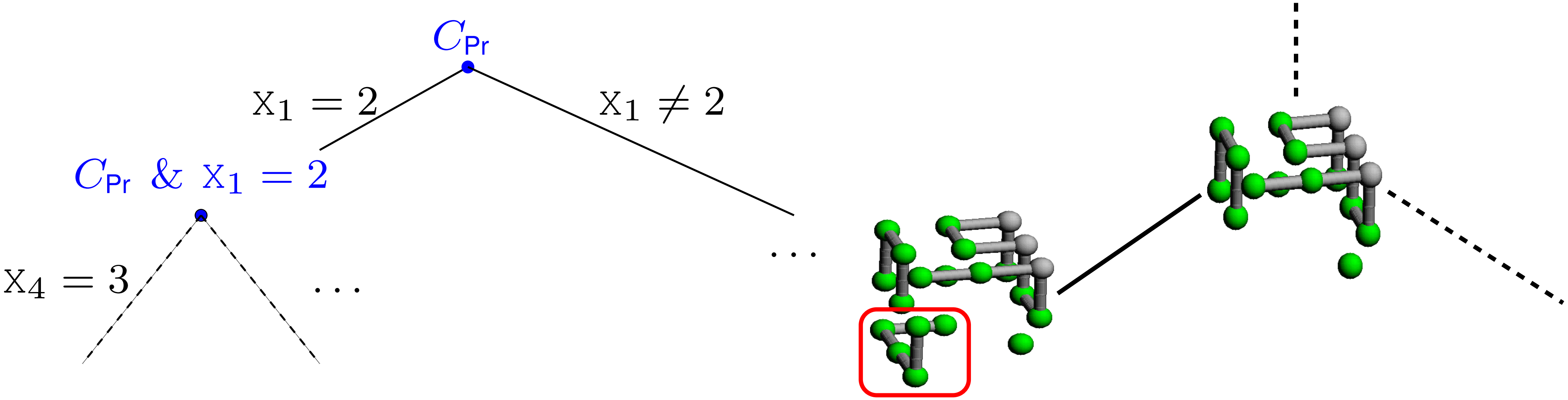


Constraint-based Formulation

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

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

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



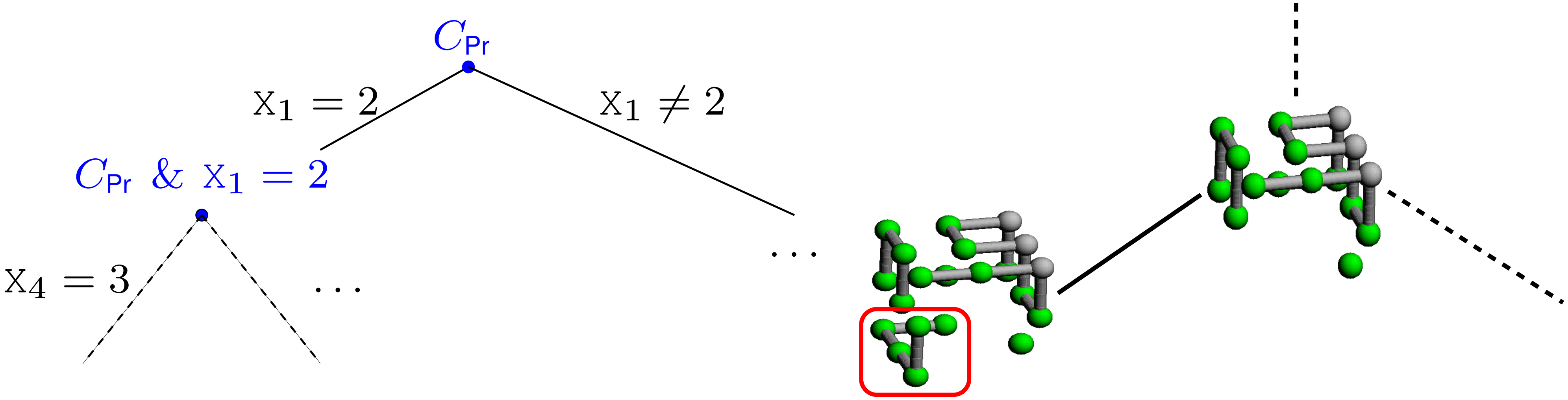
- **problems**
  - redundant constraints and search strategy [Backofen:98]
  - symmetry breaking [Backofen&Will:99]
  - bound for number of HH-contacts [Backofen:00a,03]
  - new constraints, propagation [Backofen:Will:01]

Constraint-based Formulation

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

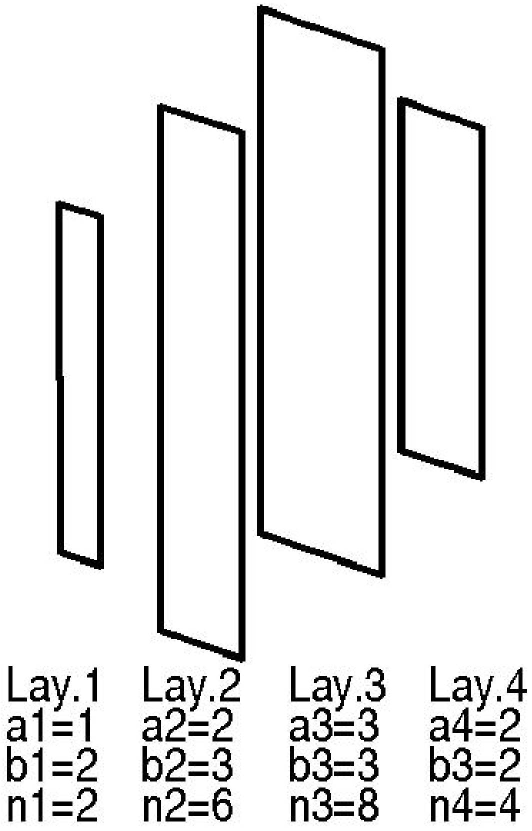
$$(\mathtt{X}_i, \mathtt{Y}_i, \mathtt{Z}_i) \neq (\mathtt{X}_j, \mathtt{Y}_j, \mathtt{Z}_j) \quad \text{and} \quad |(\mathtt{X}_i, \mathtt{Y}_i, \mathtt{Z}_i) - (\mathtt{X}_{i+1}, \mathtt{Y}_{i+1}, \mathtt{Z}_{i+1})| = 1$$

- **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]

Problem 1: Frame Sequences



Bounds for FCC

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

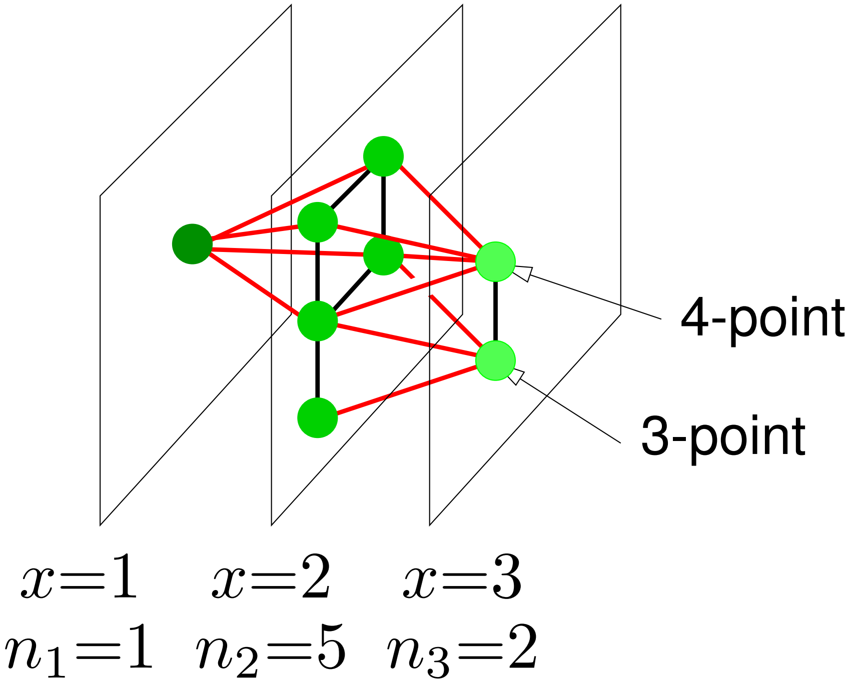
- **BUT**: almost nothing was known
  - approximation: 60% of optimum [Agarwala et al.98]
  - only trivial bounds: 6 $\times$ number of H-amino acids.

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 × number of H-amino acids.

- approach:



layer contacts

interlayer contacts

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 × 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

$$\underset{\substack{\uparrow \\ \text{number of} \\ \text{H-neighbours}}}{4n} = 2\cdot\text{H-contacts} + \underset{\substack{\uparrow \\ \text{contacts to Ps} \\ \text{or solution positions}}}{\text{H-surface}}$$

- relation to frame

Bound on Layer Contact

- relation between surface and contacts

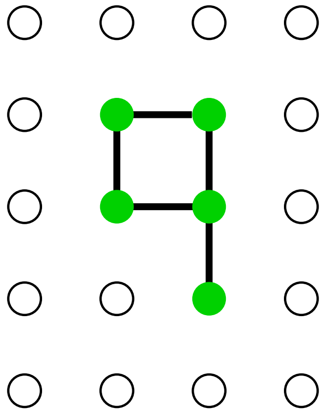
$$4n = 2 \cdot \text{H-contacts} + \text{H-surface}$$

↑ (under 4n): number of H-neighbours

↑ (under H-surface): contacts to Ps or solution positions

- relation to frame

horizontal surface



Bound on Layer Contact

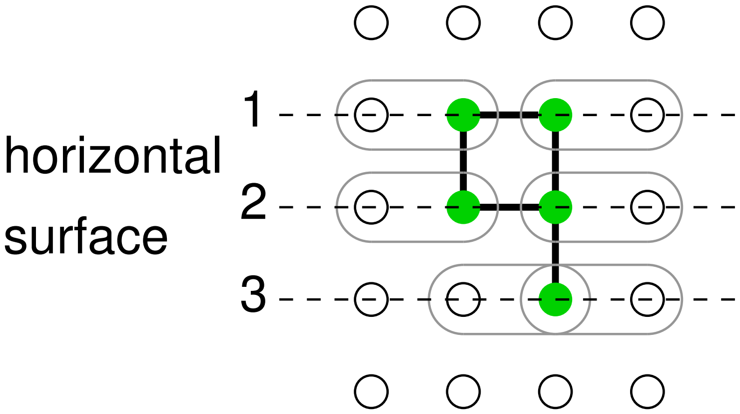
- relation between surface and contacts

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

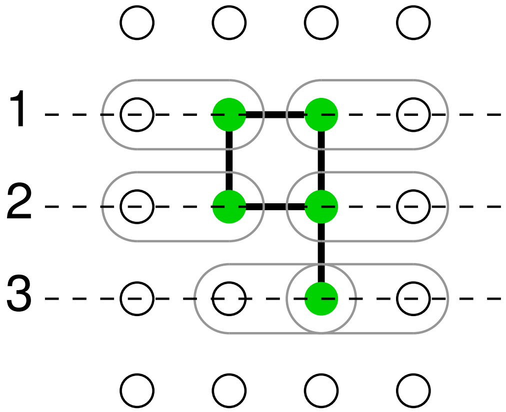
4n = 2·H-contacts + H-surface

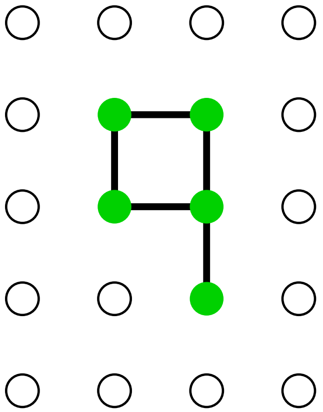
↑ ↑

number of H-neighbours — contacts to Ps or solution positions

- relation to frame

horizontal surface





vertical surface

Bound on Layer Contact

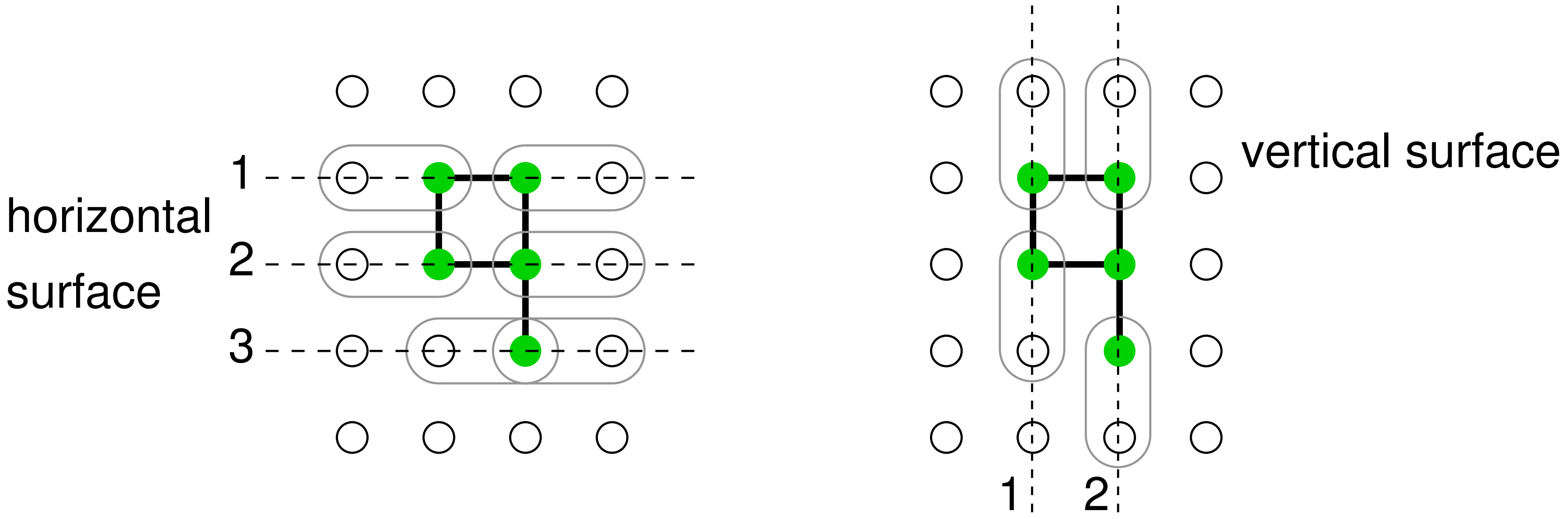
- relation between surface and contacts

4n = 2·H-contacts + H-surface

↑ number of H-neighbours

↑ contacts to Ps or solution positions

- relation to frame



Bound on Layer Contact

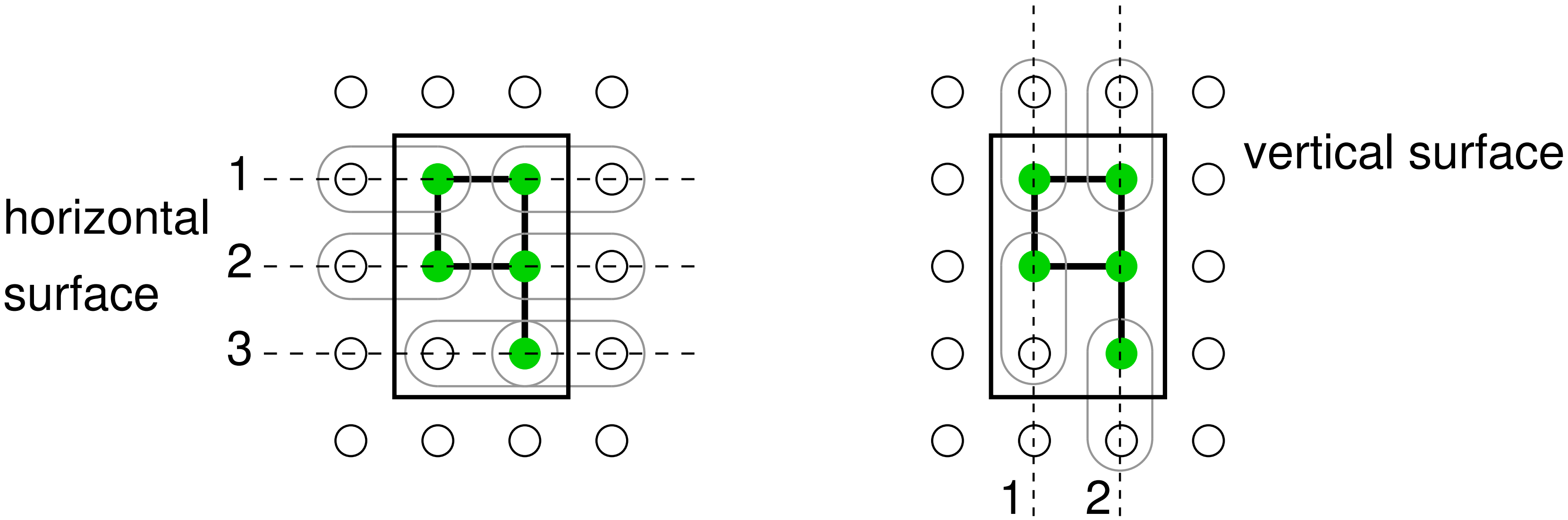
- relation between surface and contacts

4n = 2·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

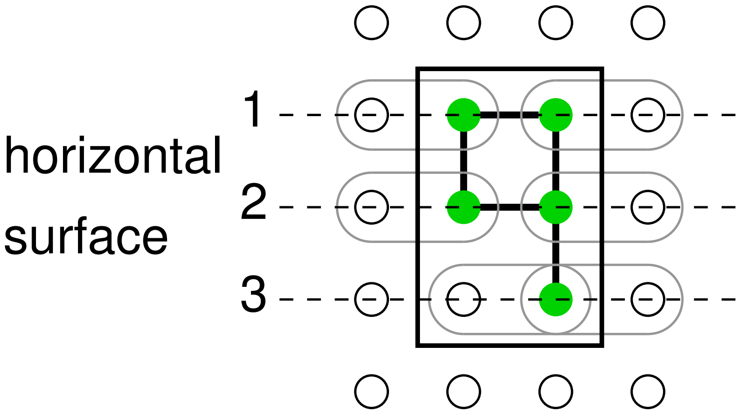
- relation between surface and contacts

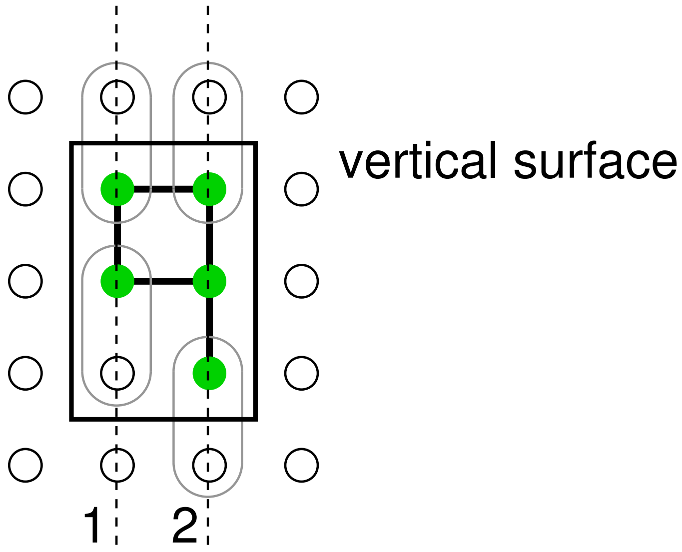
4n = 2·H-contacts + H-surface

↑ 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)





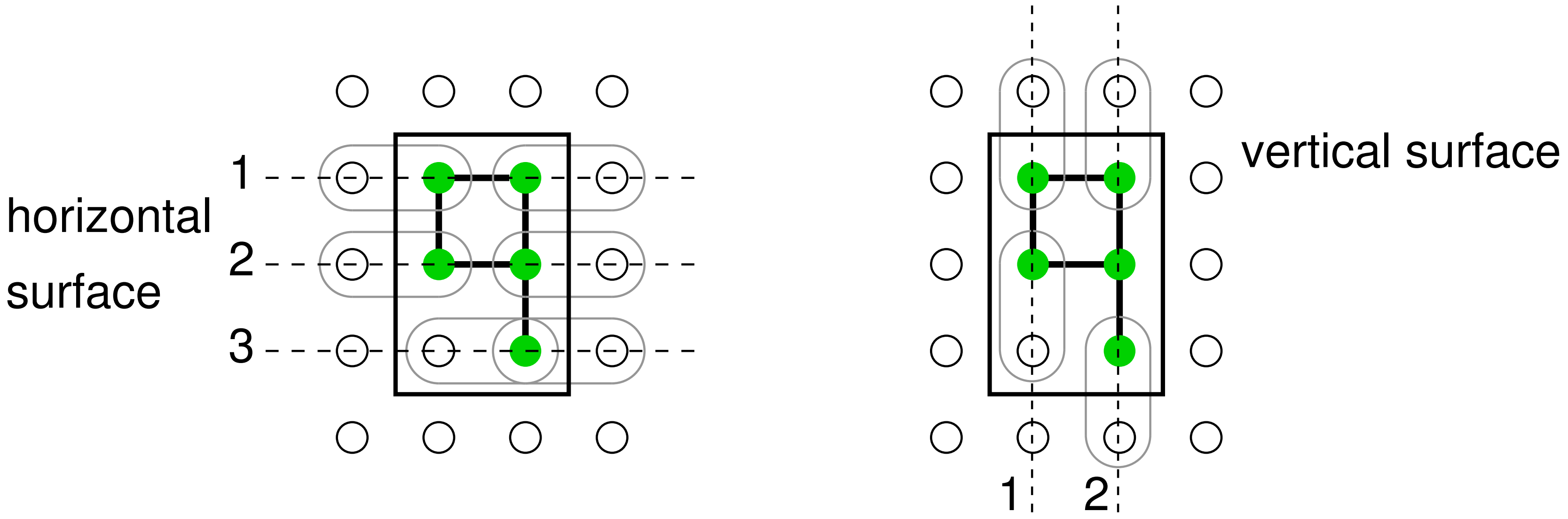
- minimal surface for n Hs $=$ minimal frame (a, b) around n point

Bound on Layer Contact

- relation between surface and contacts

$$\underset{\substack{\uparrow\\ \text{number of}\\ \text{H-neighbours}}}{4n} = 2\cdot\text{H-contacts} + \underset{\substack{\uparrow\\ \text{contacts to Ps}\\ \text{or solution positions}}}{\text{H-surface}}$$

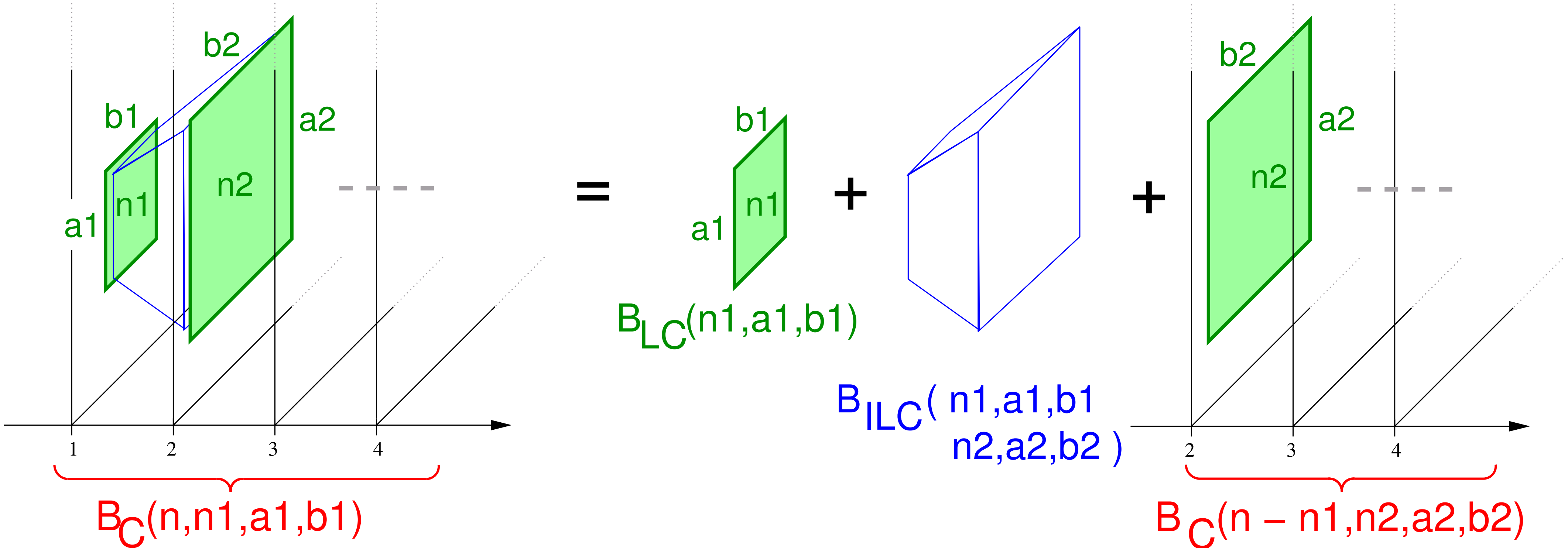
- relation to frame H-surface $= 2 \cdot a + 2 \cdot b$ (a, b)= (height,width)



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

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

Recursion for Bound

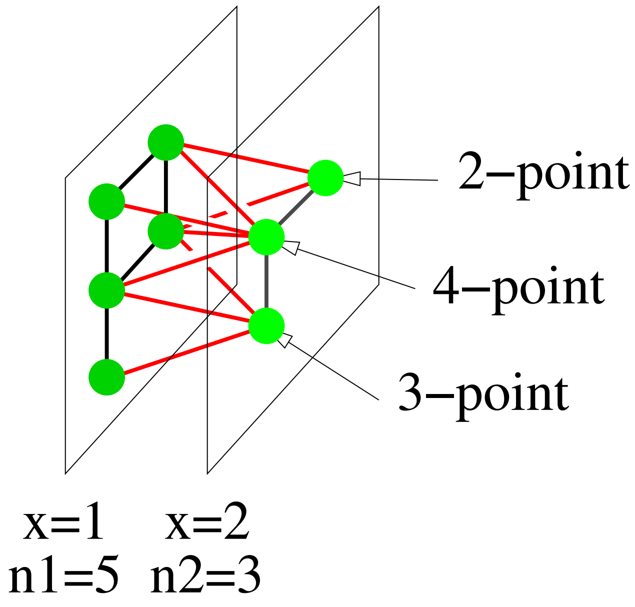


$\mathrm{B}_{\mathrm{C}}(n, n_1, a_1, b_1)$: contacts in core with n elements and first layer $E_1 : n_1, a_1, b_1$

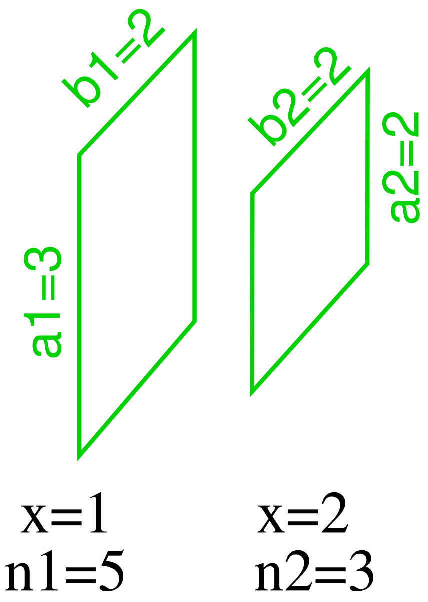
=	$\mathrm{B}_{\mathrm{LC}}(n_1, a_1, b_1)$	contacts in layer E_1
+	$\mathrm{B}_{\mathrm{ILC}}(n_1, a_1, b_1, n_2, a_2, b_2)$	contacts between layers E_1 and $E_2 : n_2, a_2, b_2$
+	$\mathrm{B}_{\mathrm{C}}(n - n_1, n_2, a_2, b_2)$	contacts in core with $n - n_1$ elements and first layer E_2

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

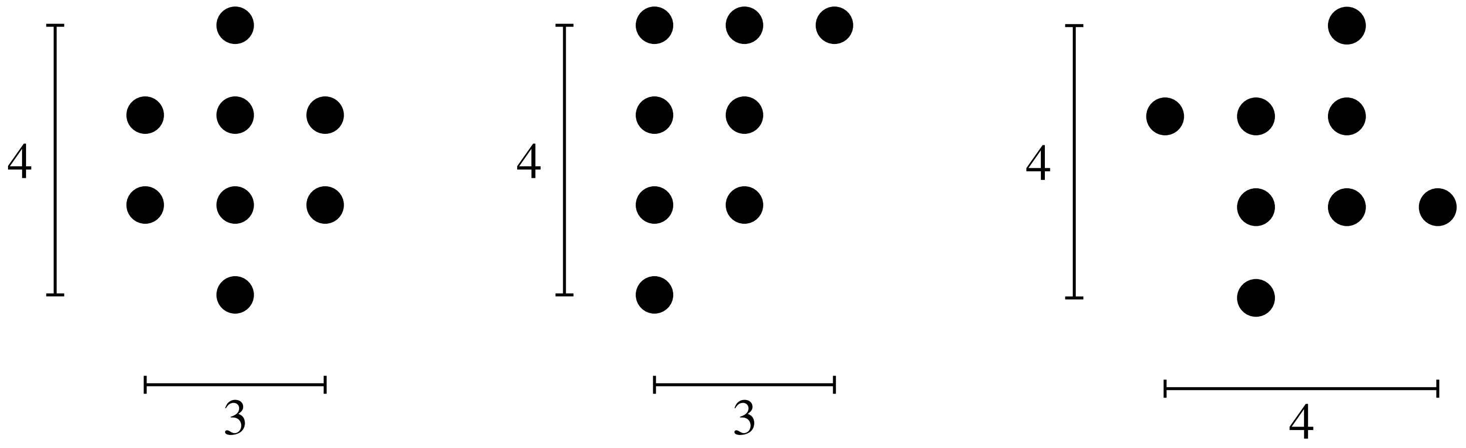
Bound on Number of 4-, 3-, 2- and 1-Points

- 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

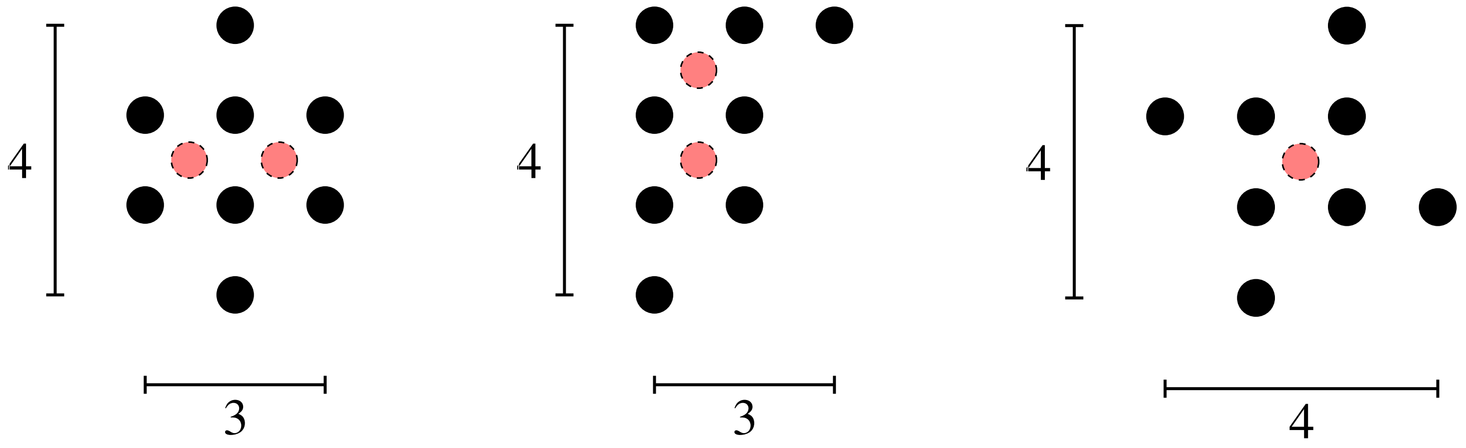
Bound on Number of 4-, 3-, 2- and 1-Points

- 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

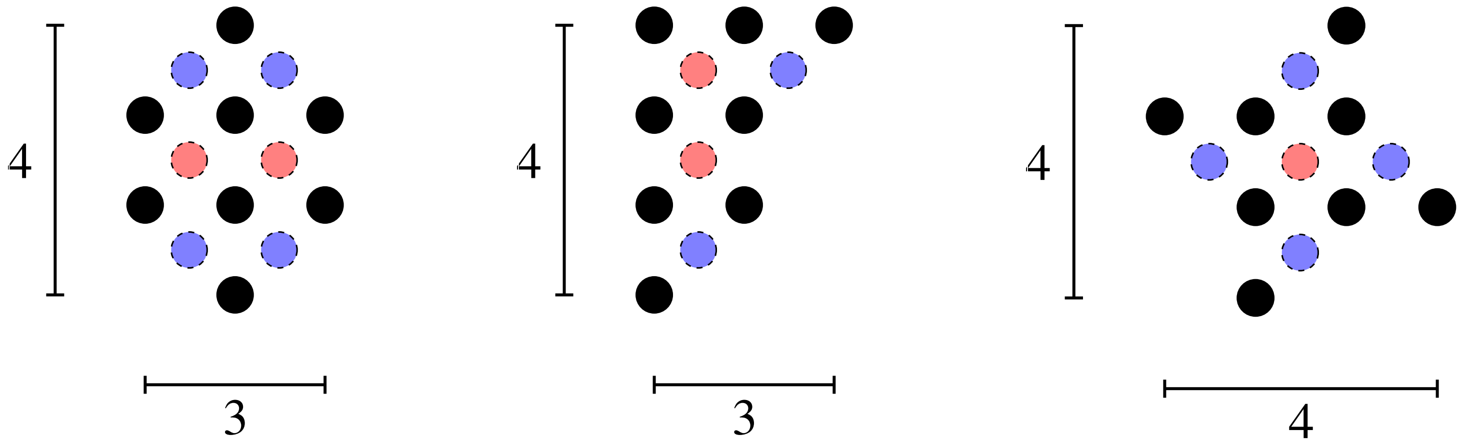
Bound on Number of 4-, 3-, 2- and 1-Points

- 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

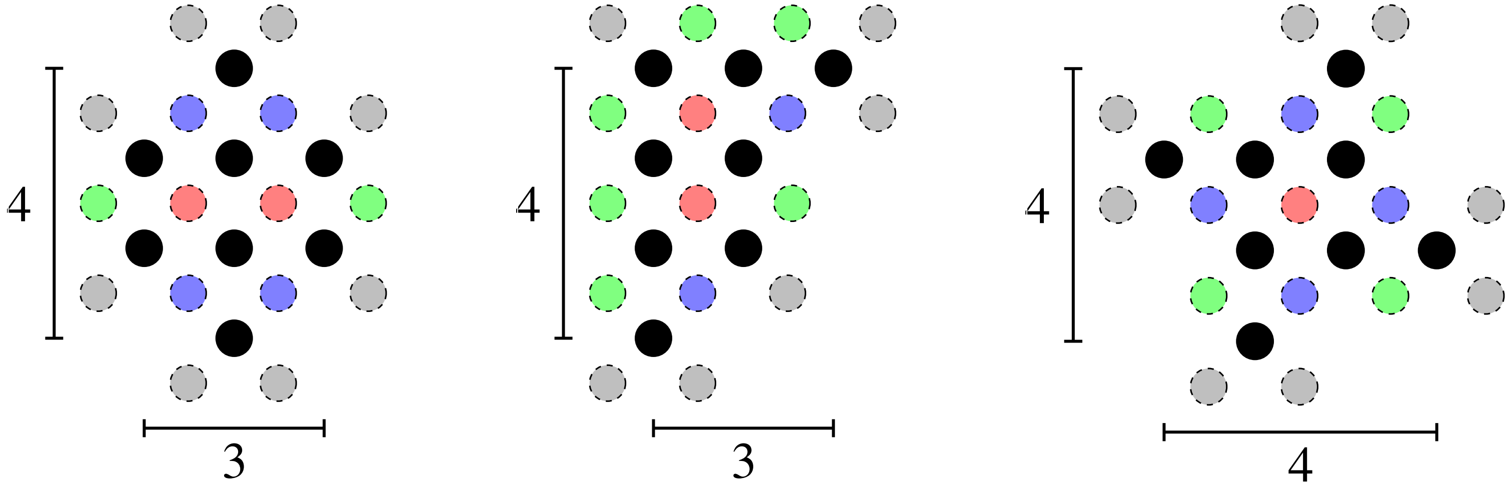
Bound on Number of 4-, 3-, 2- and 1-Points

- 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

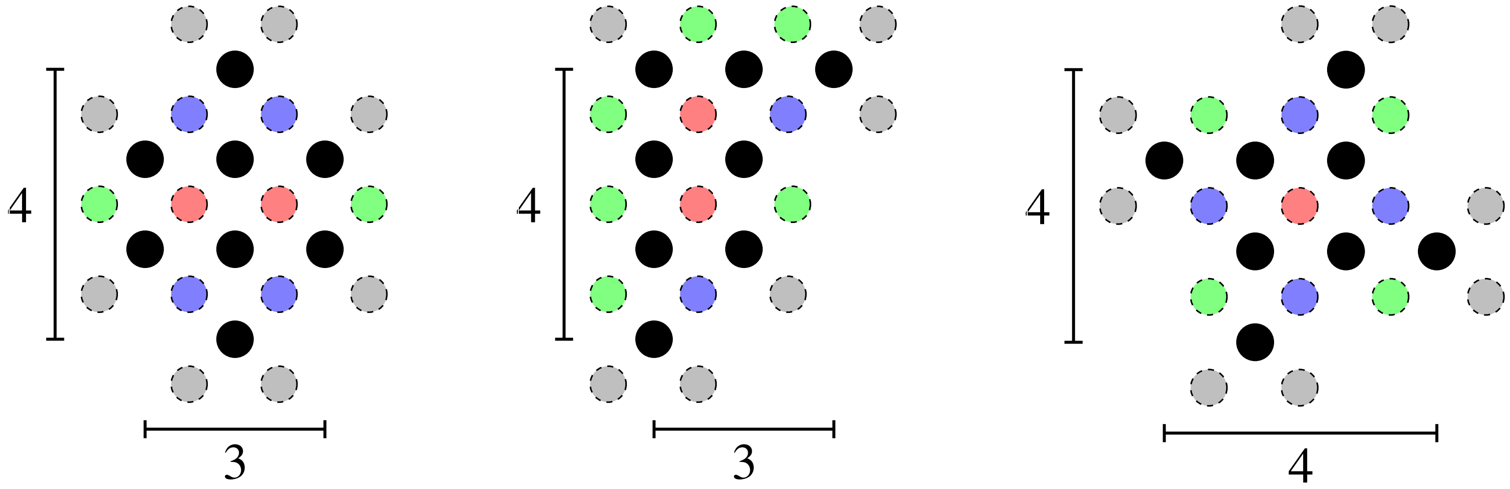
Bound on Number of 4-, 3-, 2- and 1-Points

- 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 ℓ be the number of 3-points. Then:

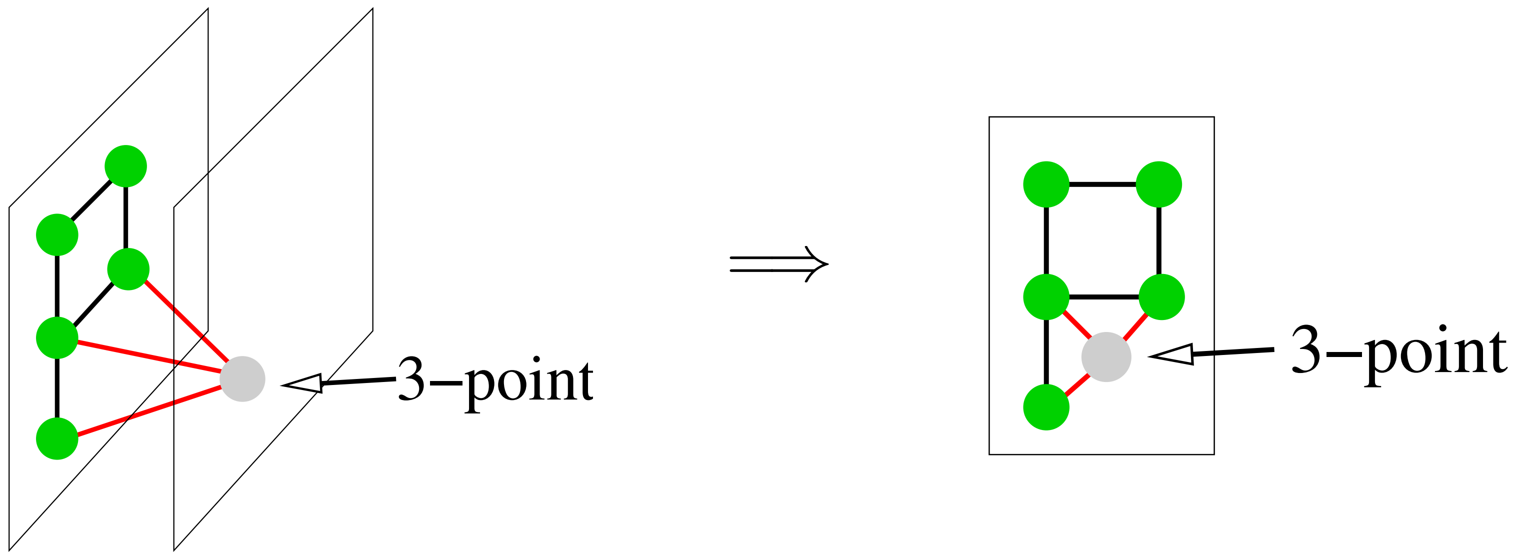
number of 4 $= n_i + 1 - a_i - b_i$ number of 2 $= 2a_i + 2b_i - 2\ell - 4$

number of 1 $= \ell + 4.$

for ℓ, there is an upper bound [Backofen00]

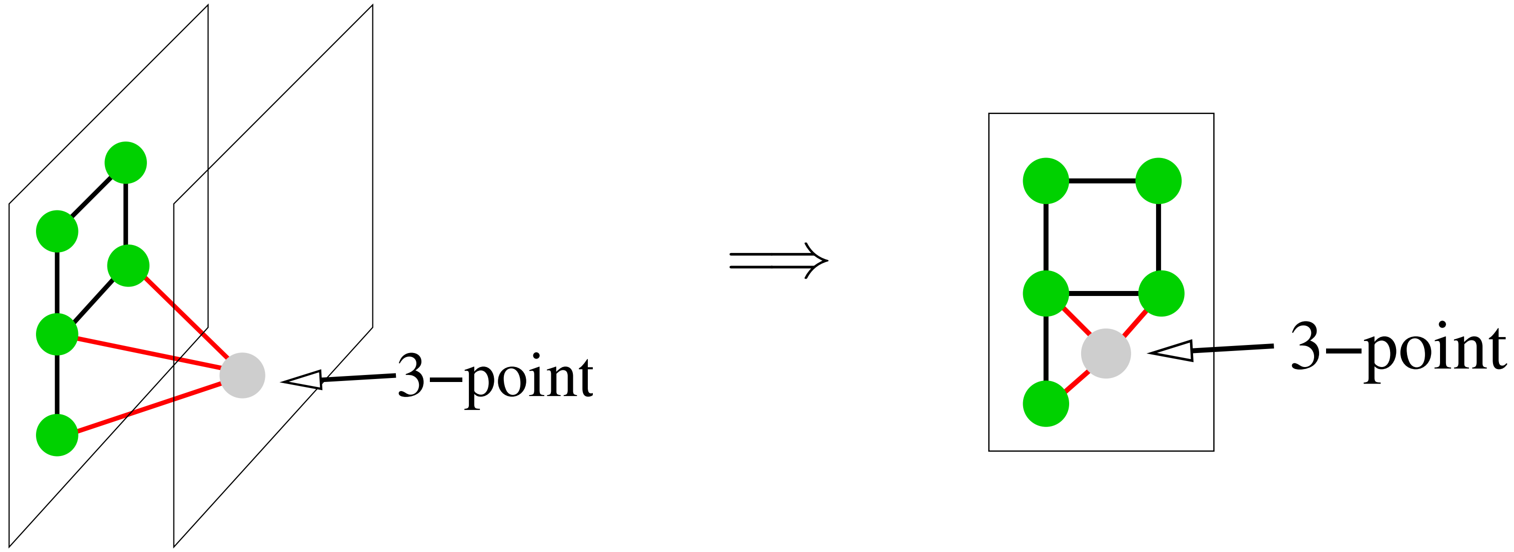
Bounds on the Number ℓ of 3-Points

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

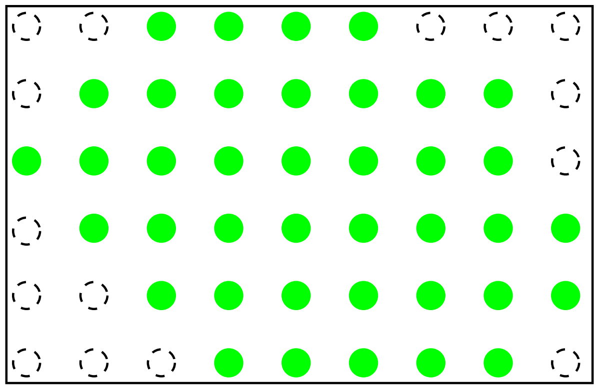


Bounds on the Number ℓ of 3-Points

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

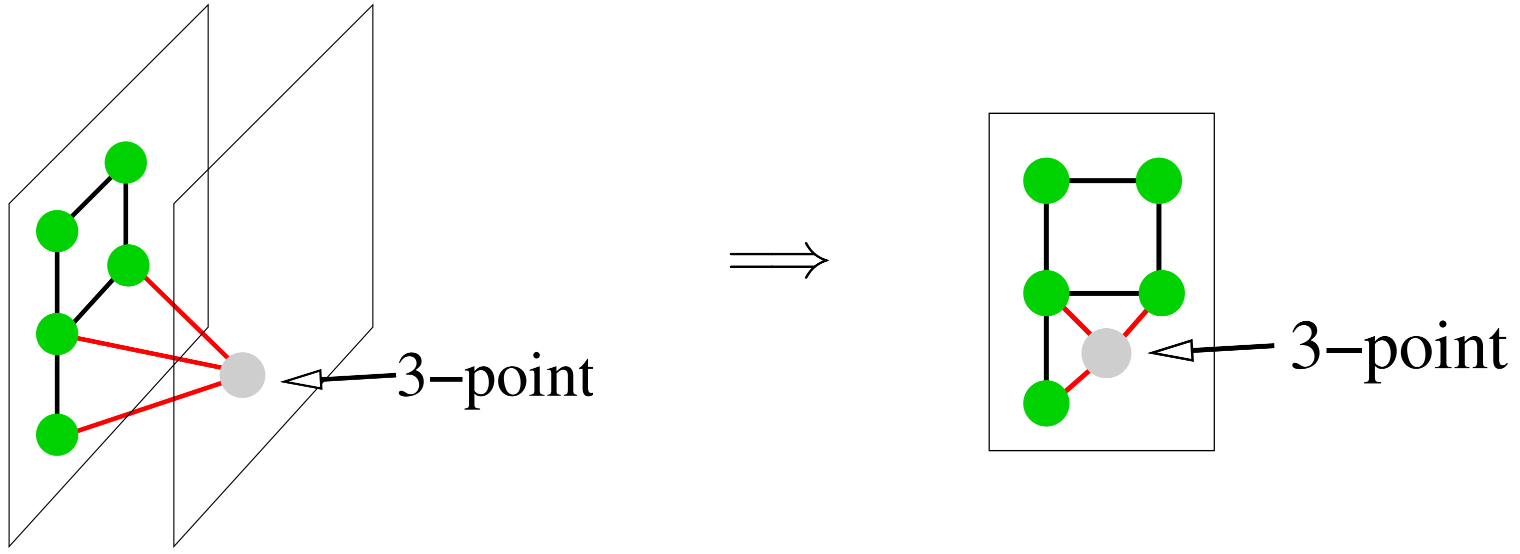


- observation: ℓ can also be calculated from the frame

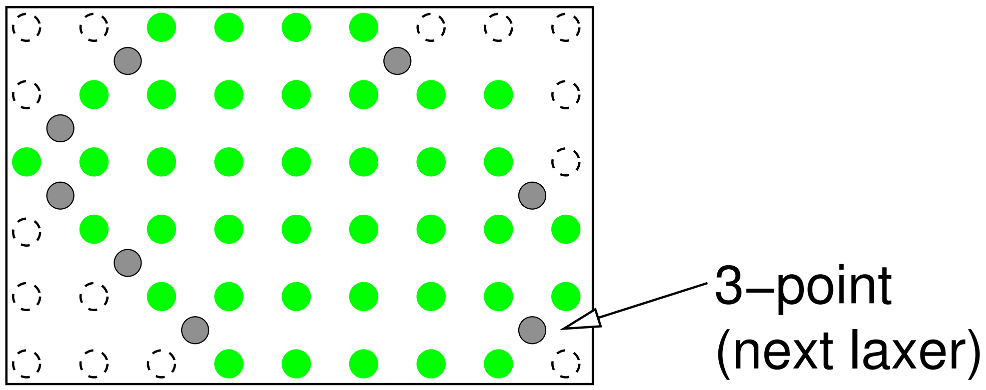


Bounds on the Number ℓ of 3-Points

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

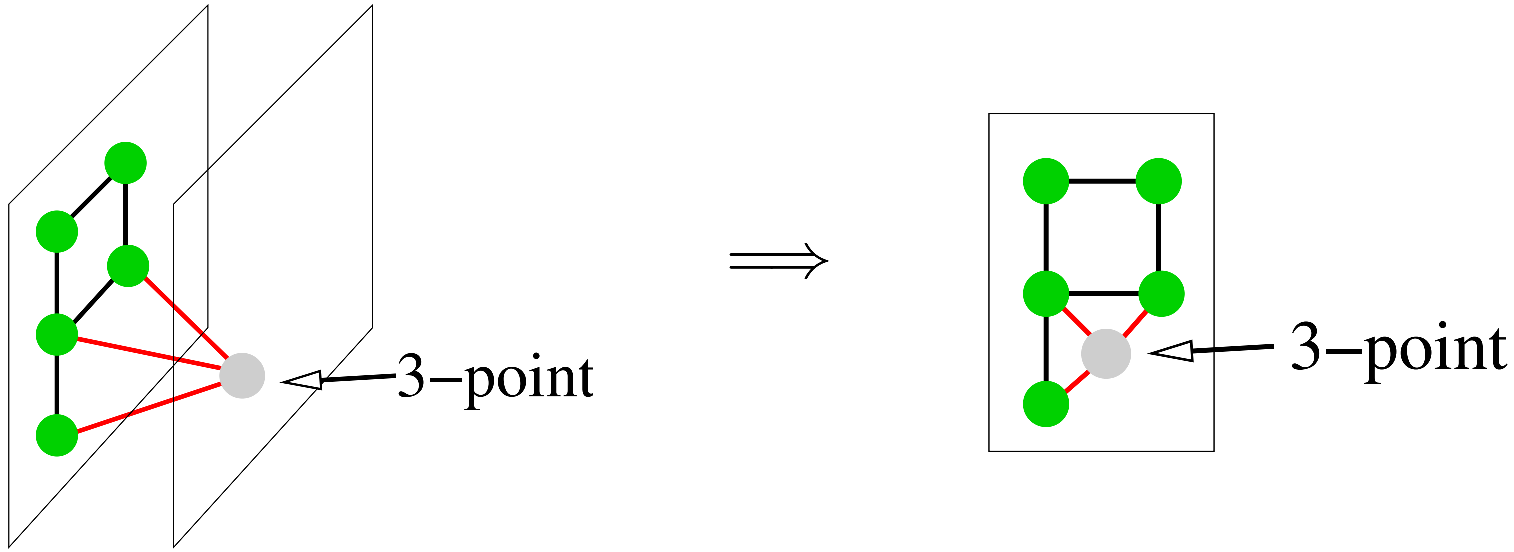


- observation: ℓ can also be calculated from the frame

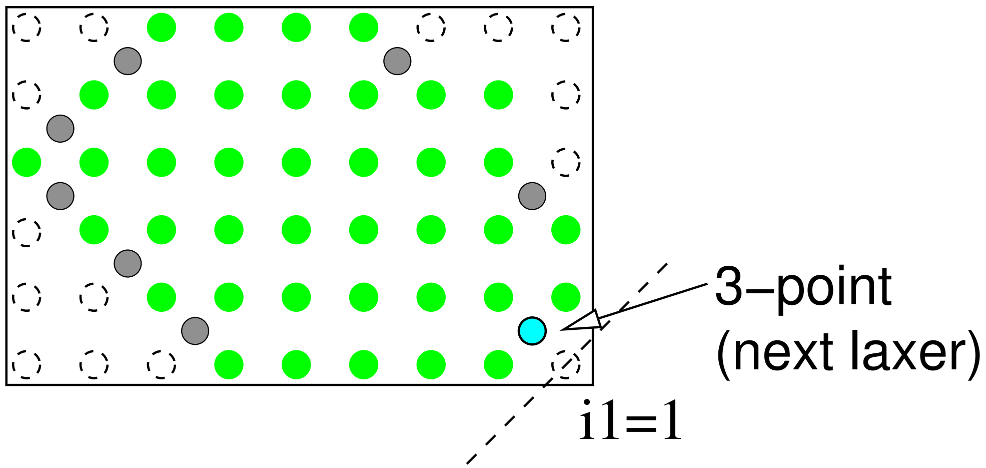


Bounds on the Number ℓ of 3-Points

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

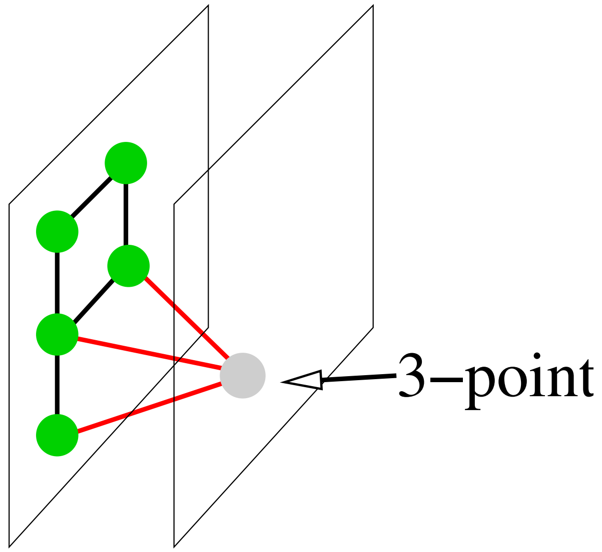


- observation: ℓ can also be calculated from the frame

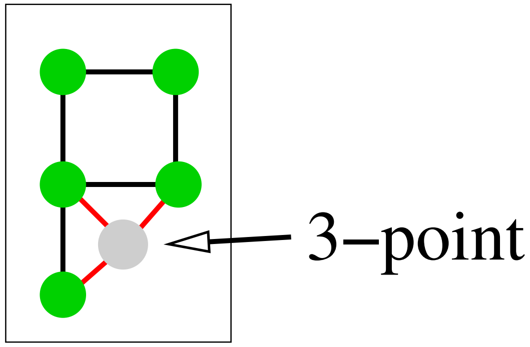


Bounds on the Number ℓ of 3-Points

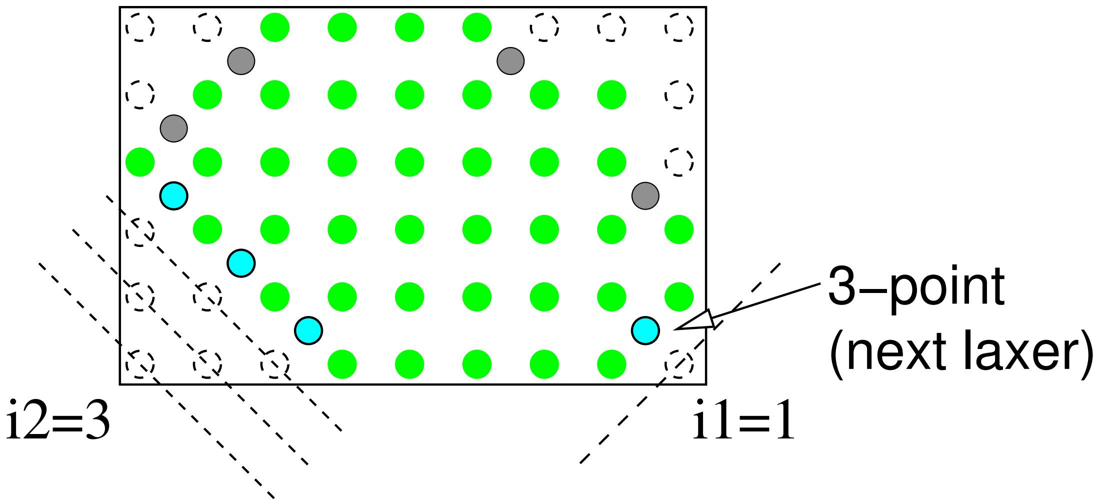
- **3-point:** from the side from the top



$\Longrightarrow$

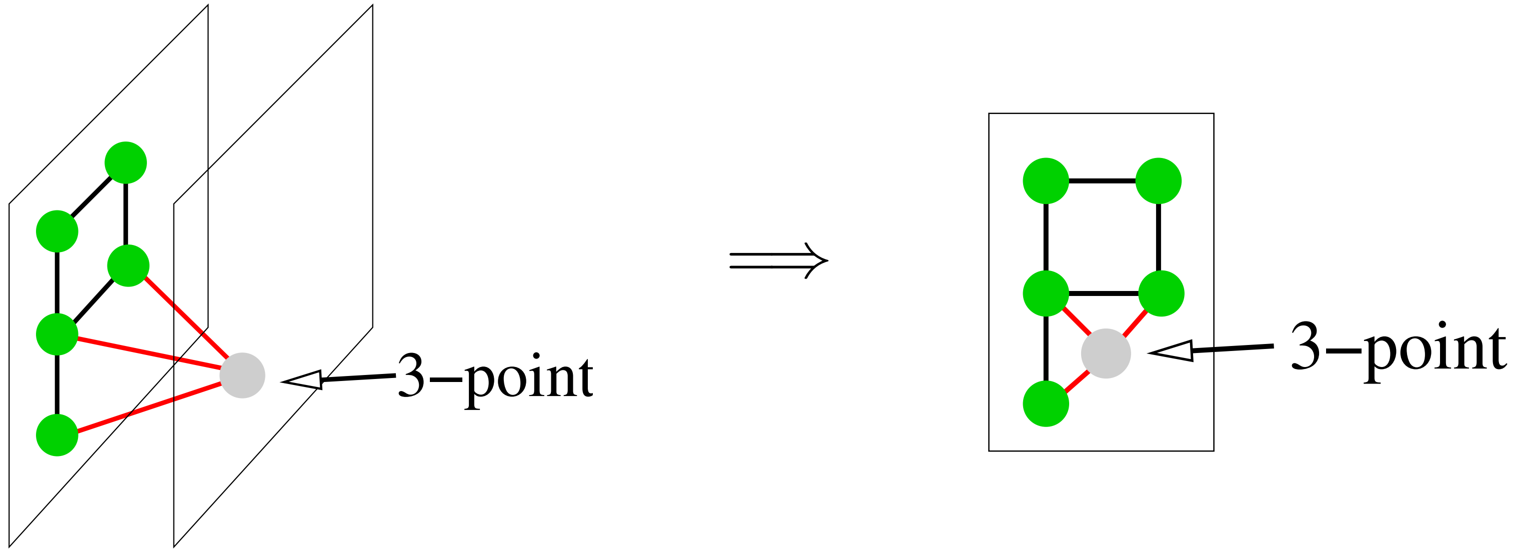


- observation: ℓ can also be calculated from the frame

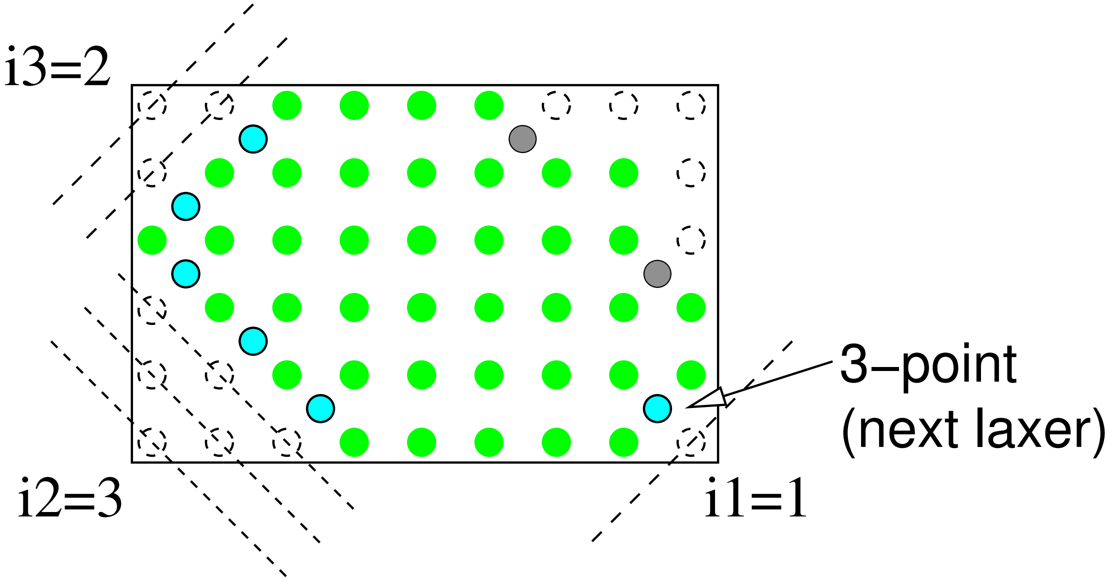


Bounds on the Number ℓ of 3-Points

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

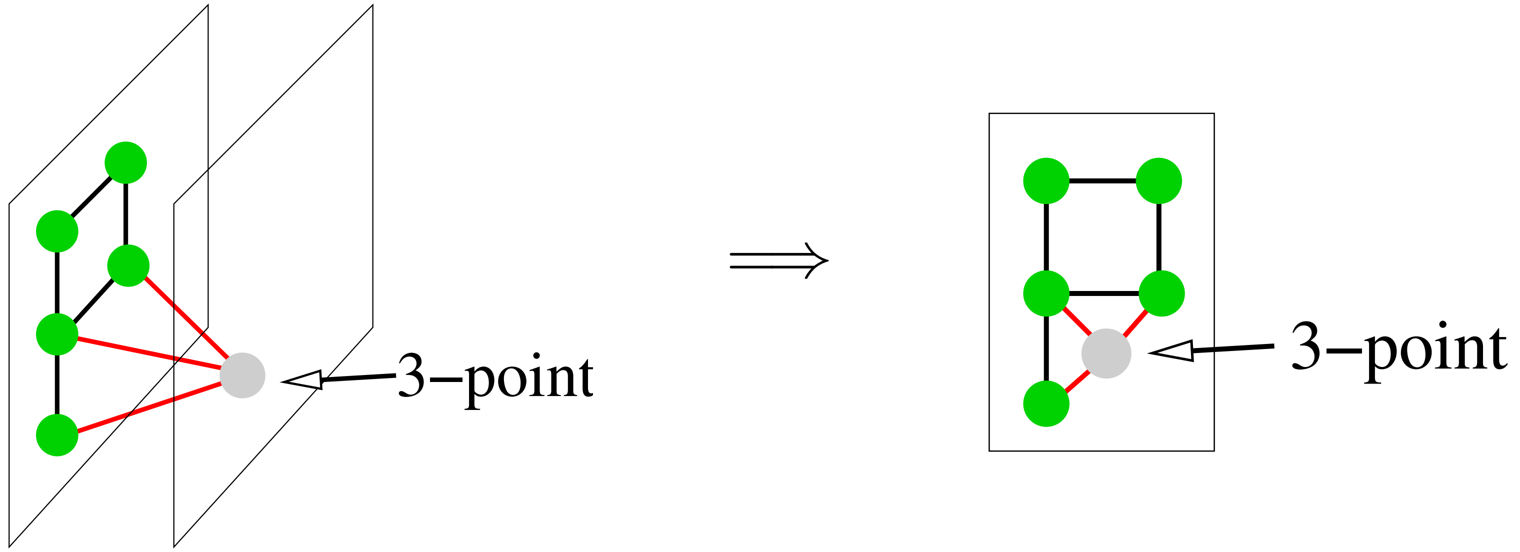


- observation: ℓ can also be calculated from the frame

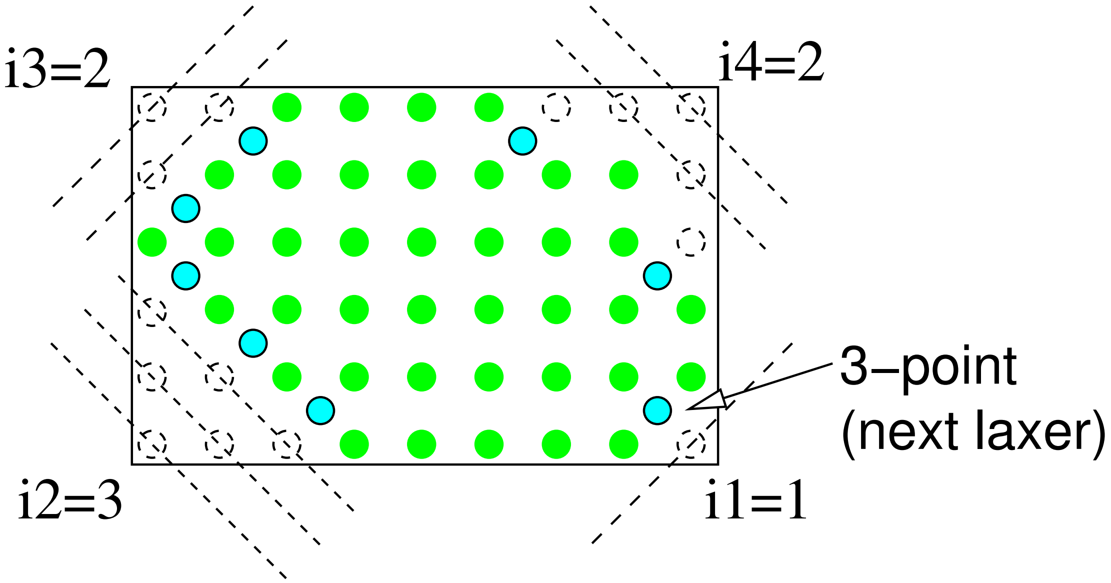


Bounds on the Number ℓ of 3-Points

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

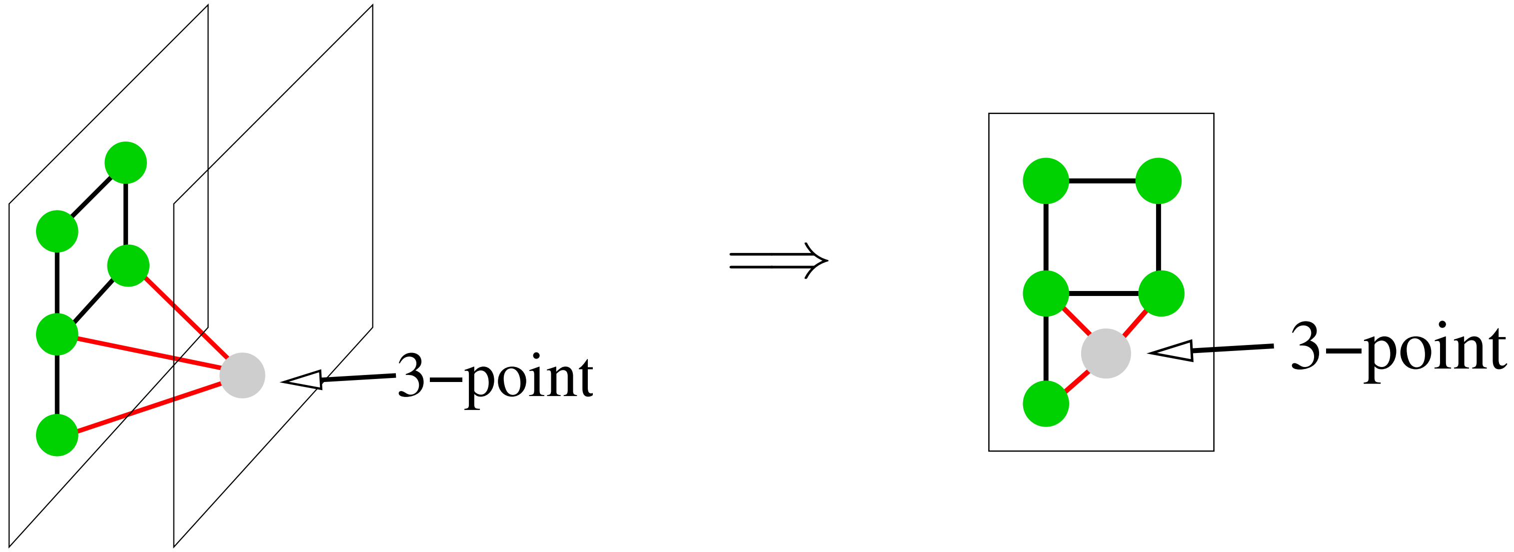


- observation: ℓ can also be calculated from the frame

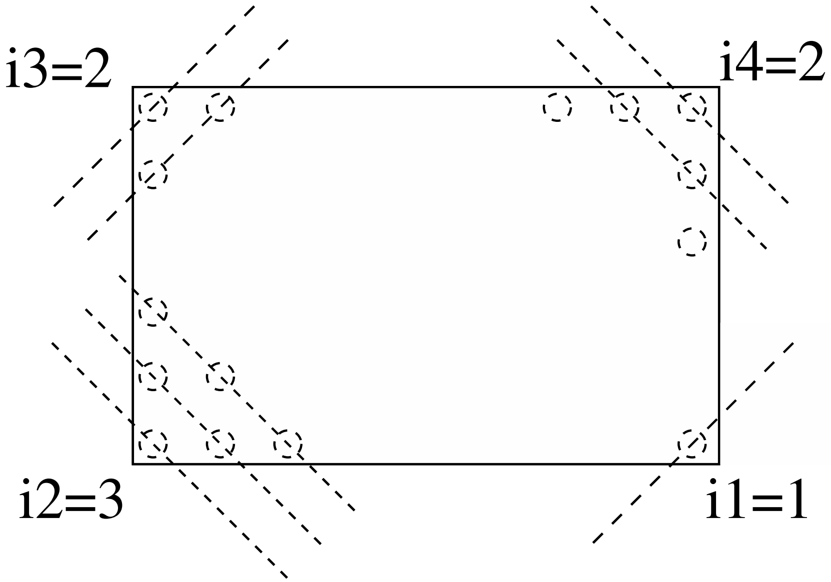


Bounds on the Number ℓ of 3-Points

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

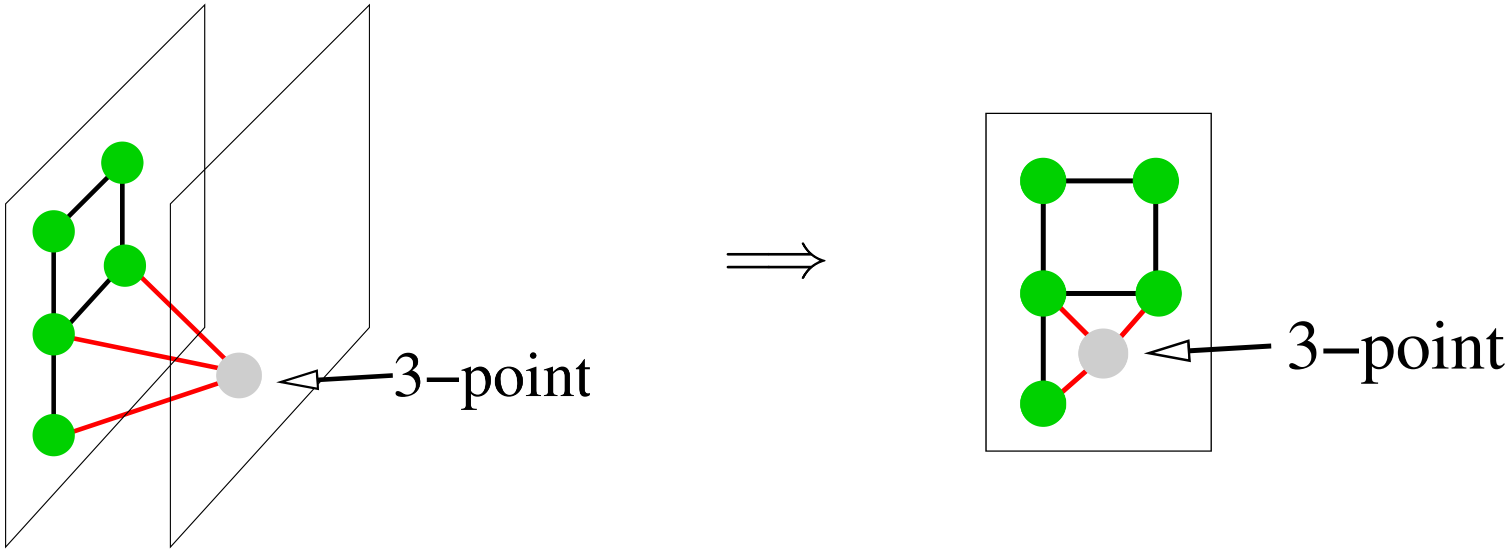


- observation: ℓ can also be calculated from the frame

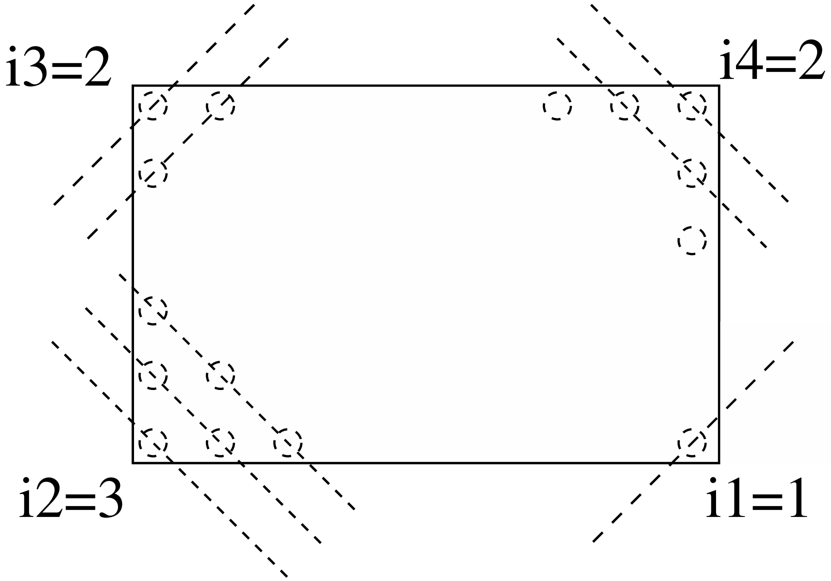


Bounds on the Number ℓ of 3-Points

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



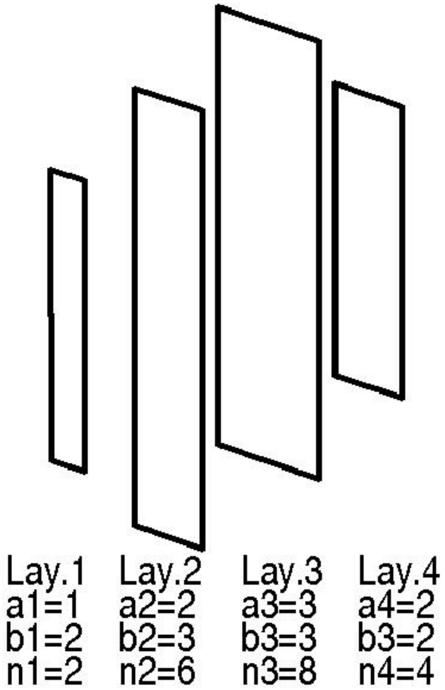
- observation: ℓ can also be calculated from the frame



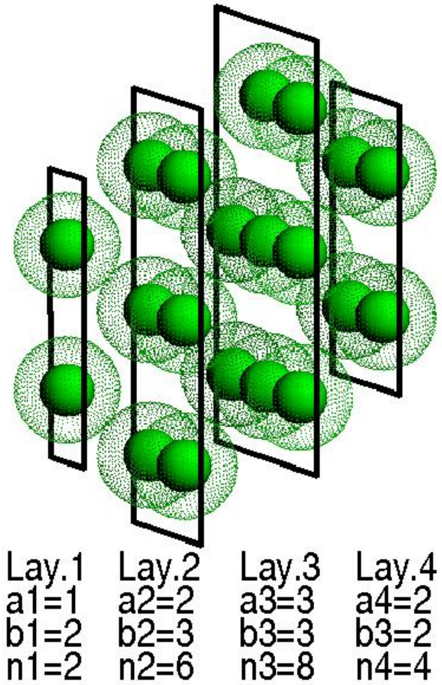
bound:

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

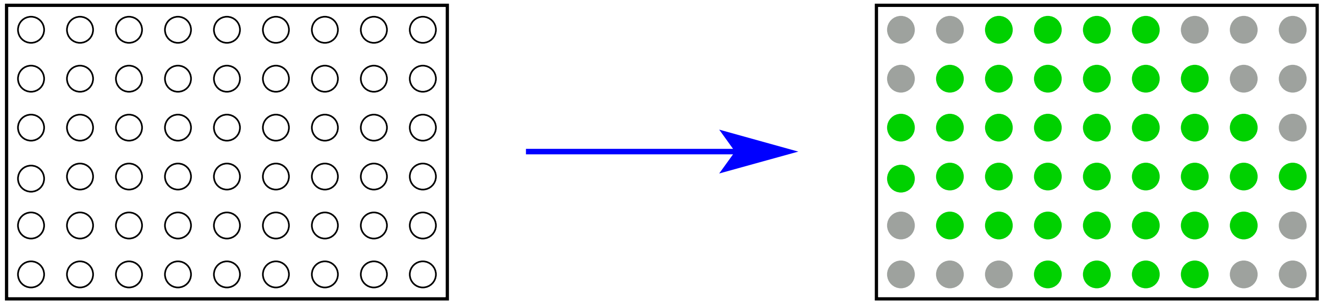
Problem 2: Enumerate Hydrophobic Cores



$\Longrightarrow$



Enumerating Hydrophobic Cores



- constraint variables:
  - boolean variable for every position

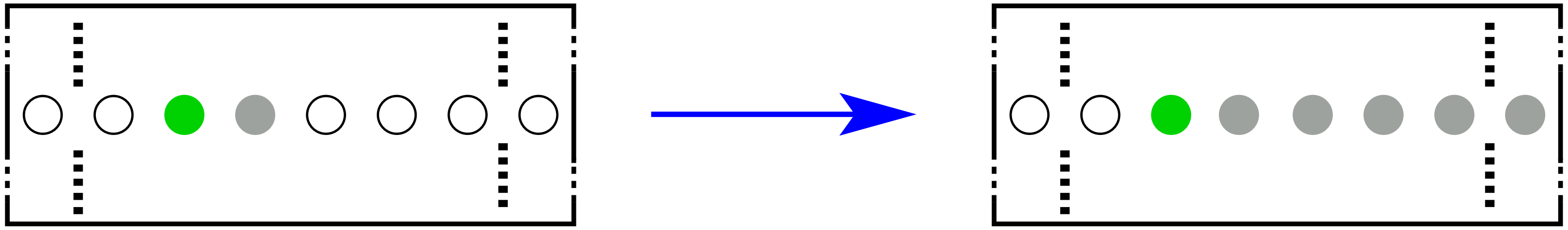
    🟢 $\hat{=}$ $(\mathbf{pnt}(\vec{p}) = 1)$

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

    ○ $\hat{=}$ $(\mathbf{pnt}(\vec{p})$ undef.$)$
  - contact variable for each neighboring position

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

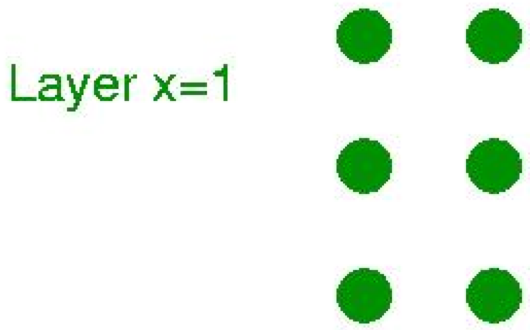
- constraints:
  - $\sum_{\vec{p}\in\mathrm{frames}} \mathbf{pnt}(\vec{p}) =$ number of Hs
  - if optimal, then no caveats

    
  - …

Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

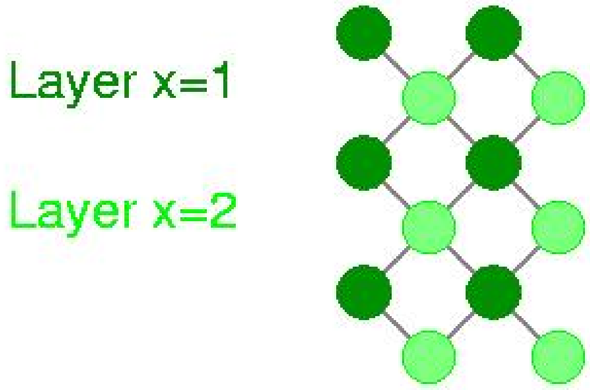
- subproblems:
    - symmetries **later**



Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

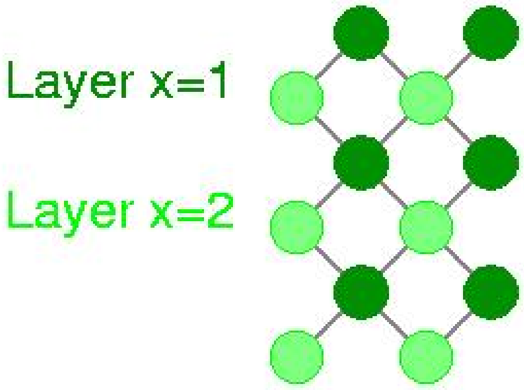
- subproblems:
  - symmetries **later**



Enumerating Hydrophobic Cores

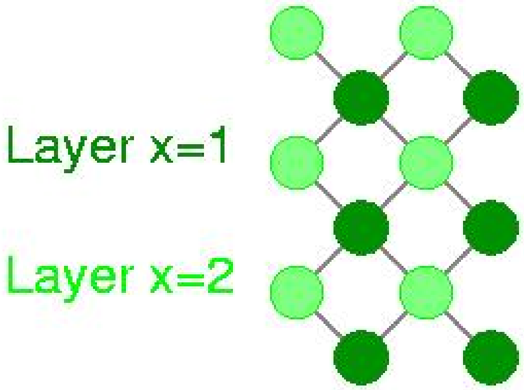
- remaining problem: relative positions of frames

- subproblems:
  - symmetries **later**



Enumerating Hydrophobic Cores

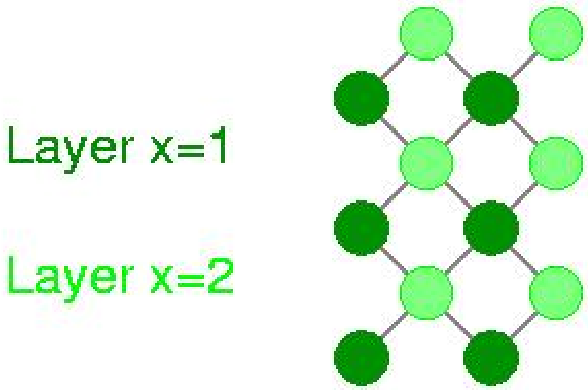
- remaining problem: relative positions of frames
- subproblems:
    - symmetries **later**



Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

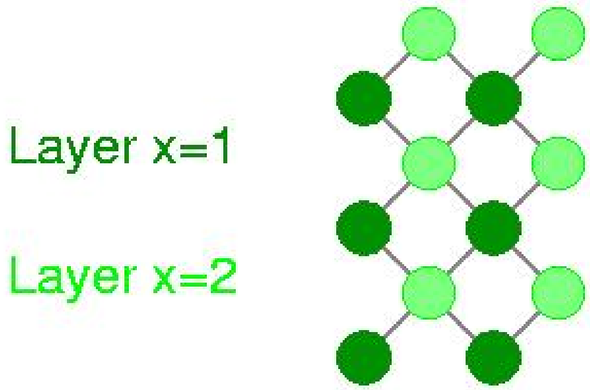
- subproblems:
  - symmetries **later**



Enumerating Hydrophobic Cores

- remaining problem: relative positions of frames

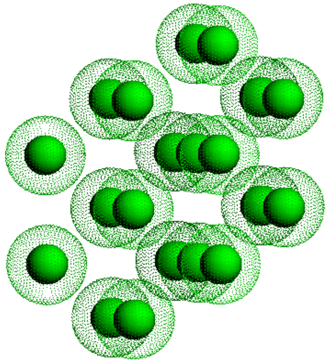
- subproblems:
  - symmetries **later**



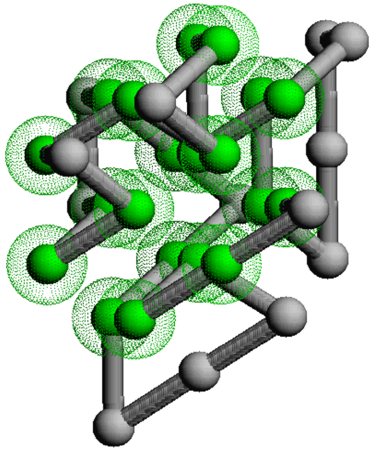
  - many subproblems solved several times
    - do not use fixed frame position
    - global bind frame positions by surrounding cube
  - more pruning: optimal core must have optimal frame-sequence in any direction

constructive disjunction

Problem 3: Threading Sequence onto Hydrophobic Cores

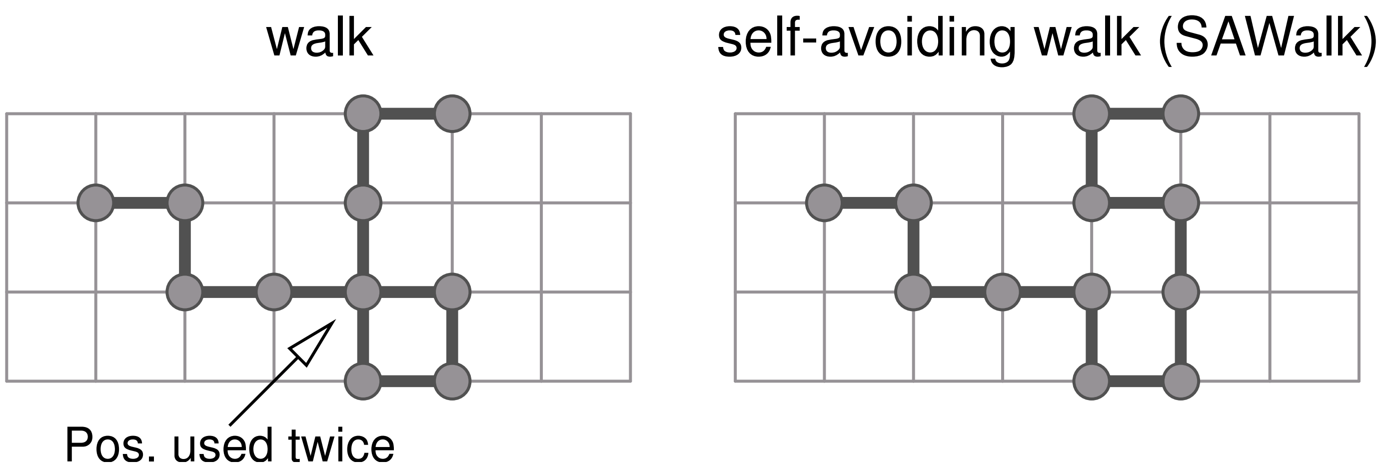


$\Longrightarrow$



New Constraints for Threading

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

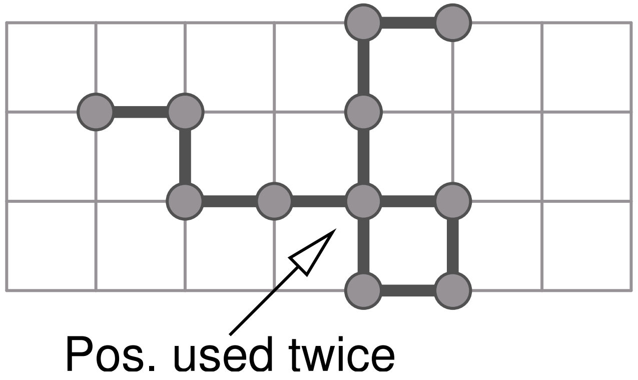


- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard

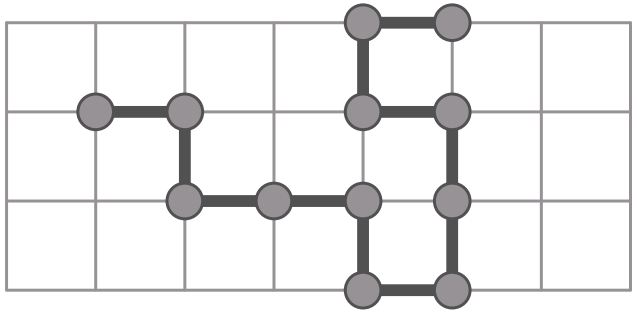
New Constraints for Threading

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

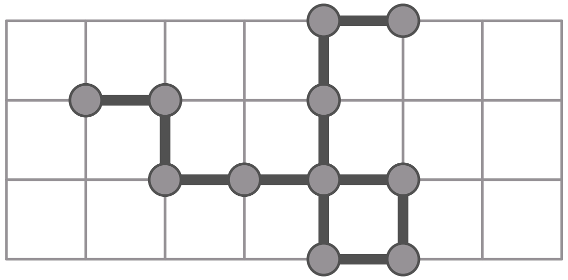
walk



self-avoiding walk (SAWalk)

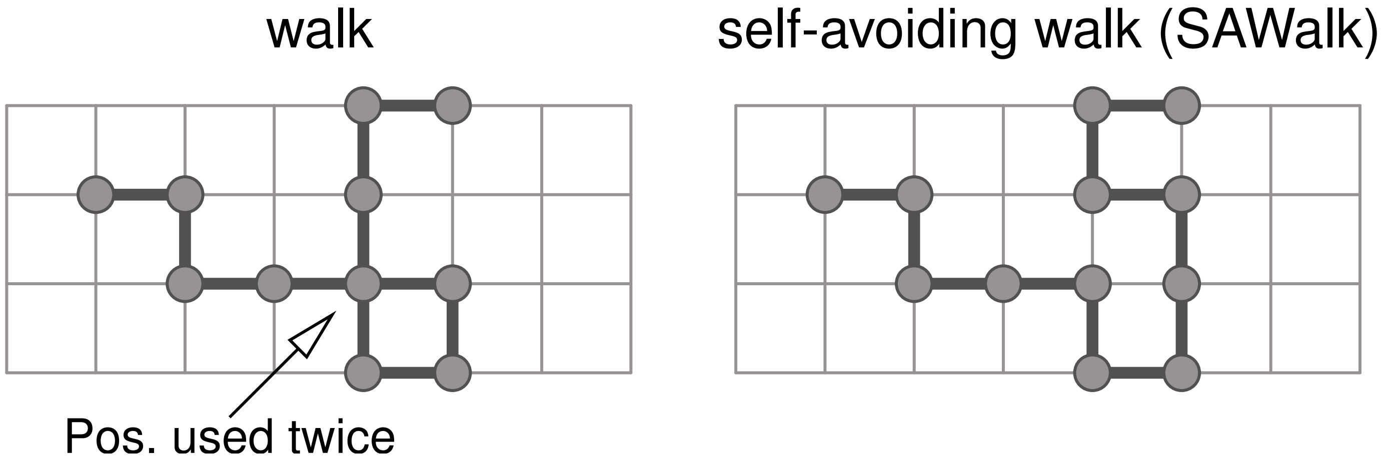


- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks $\Rightarrow$ k-avoiding walks

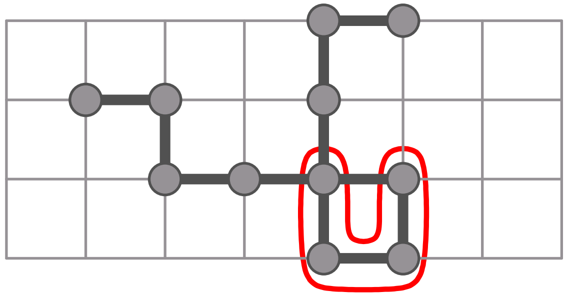


New Constraints for Threading

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



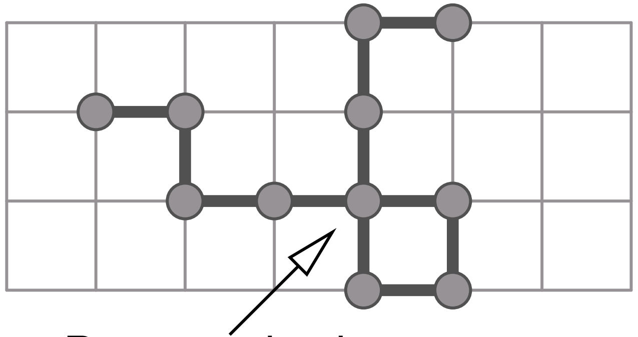
- problem: complete handling for $\mathtt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks ⇒ k-avoiding walks



New Constraints for Threading

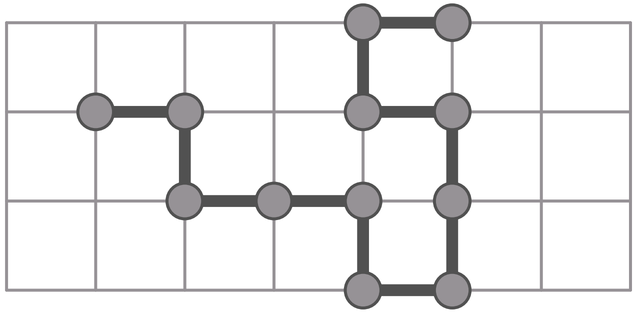
- threading: given core, find a sequence of monomer through it
- main problem: self-avoiding walks ⇒ new constraint: $\texttt{SAWalk}(x_1, \ldots, x_m)$

walk

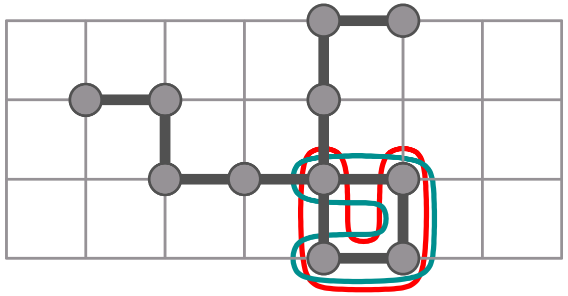


Pos. used twice

self-avoiding walk (SAWalk)

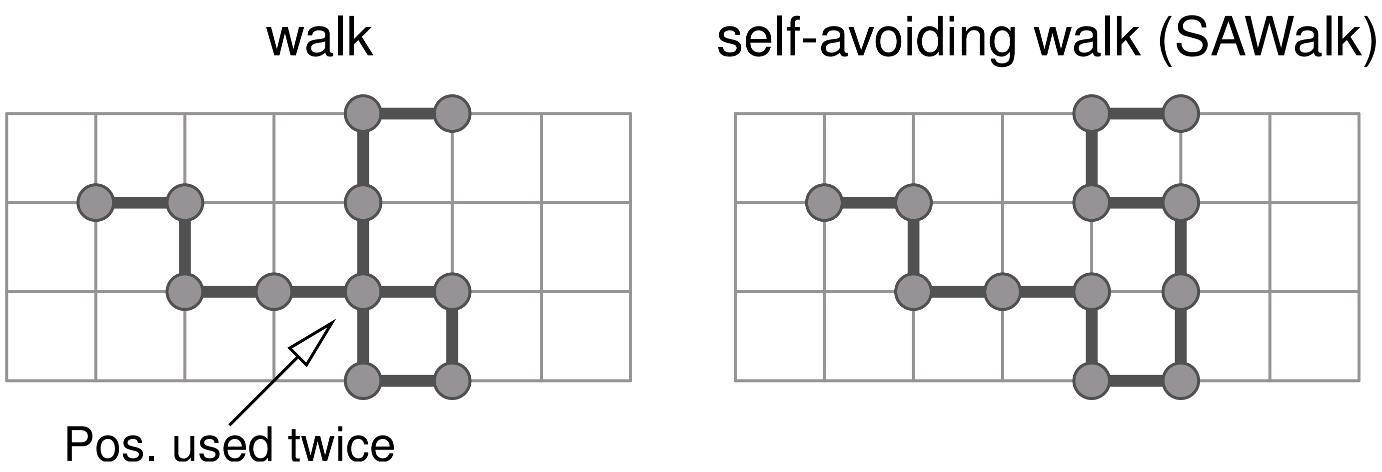


- problem: complete handling for $\texttt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks ⇒ k-avoiding walks

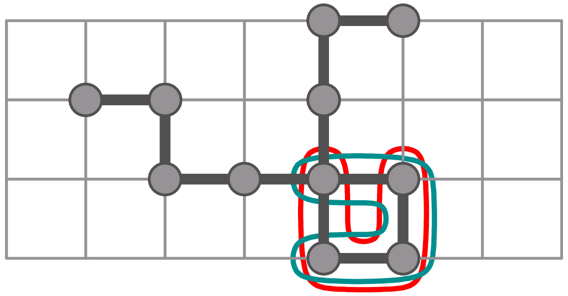


New Constraints for Threading

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



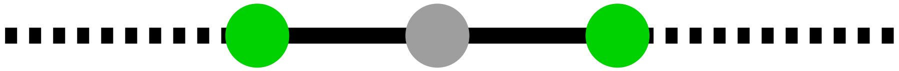
- problem: complete handling for $\mathtt{SAWalk}(x_1, \ldots, x_m)$ is hard
- therefore: approximate SAWalks ⇒ k-avoiding walks



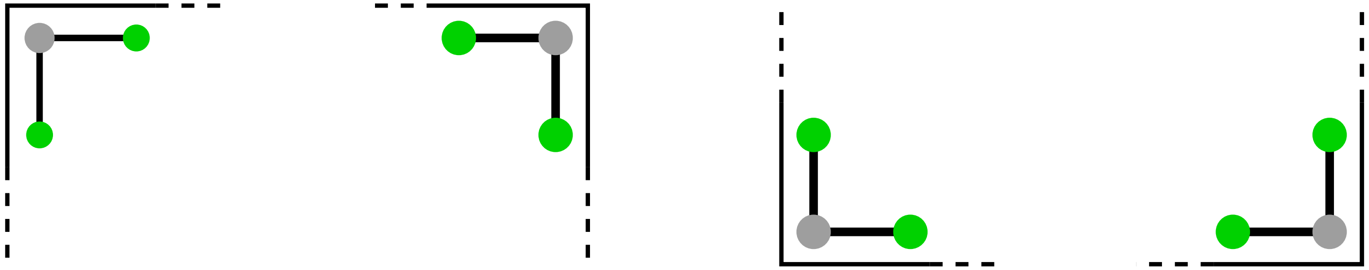
4- but not 5-avoiding

Example: 3-Avoiding

- psinglets: HPH-subsequence



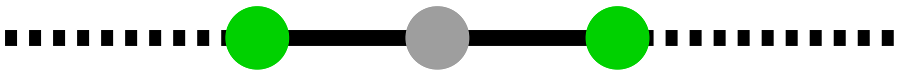
- in cubic lattice: has strong influence on core



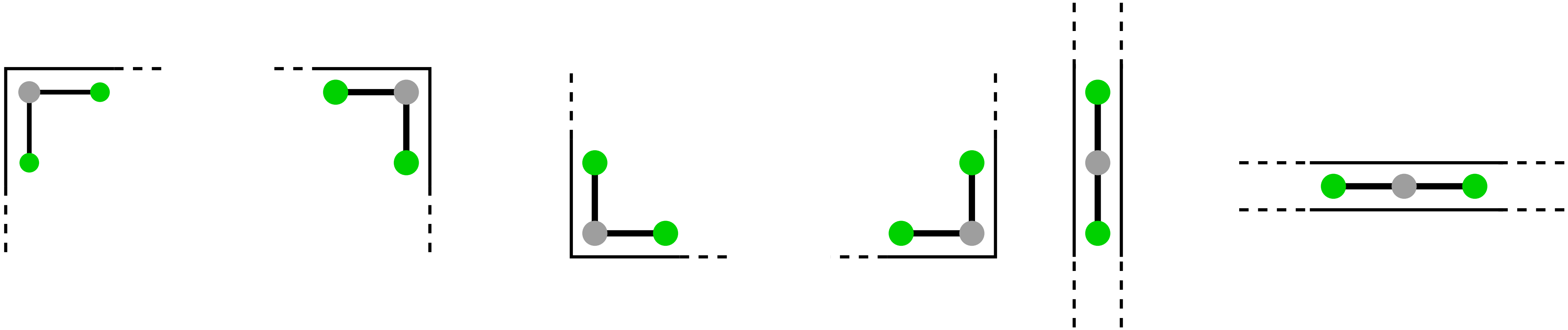
- caveat-freeness by path constraint

Example: 3-Avoiding

- psinglets: HPH-subsequence



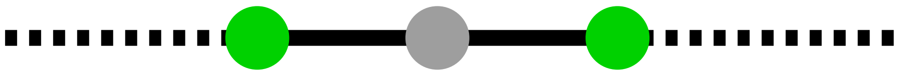
- in cubic lattice: has strong influence on core



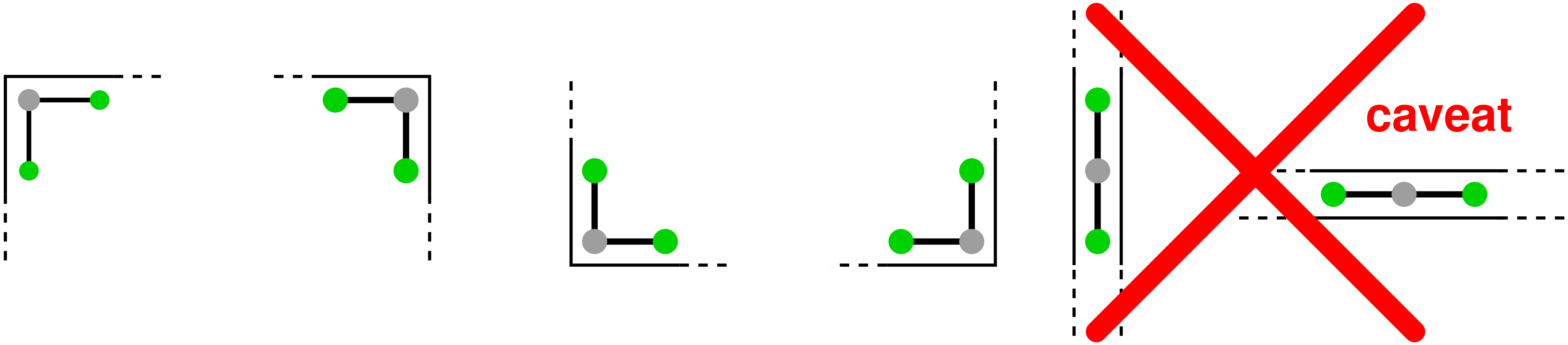
- caveat-freeness by path constraint

Example: 3-Avoiding

- psinglets: HPH-subsequence



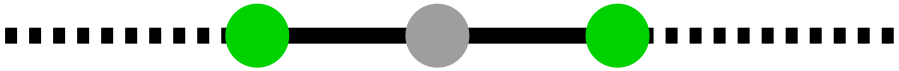
- in cubic lattice: has strong influence on core



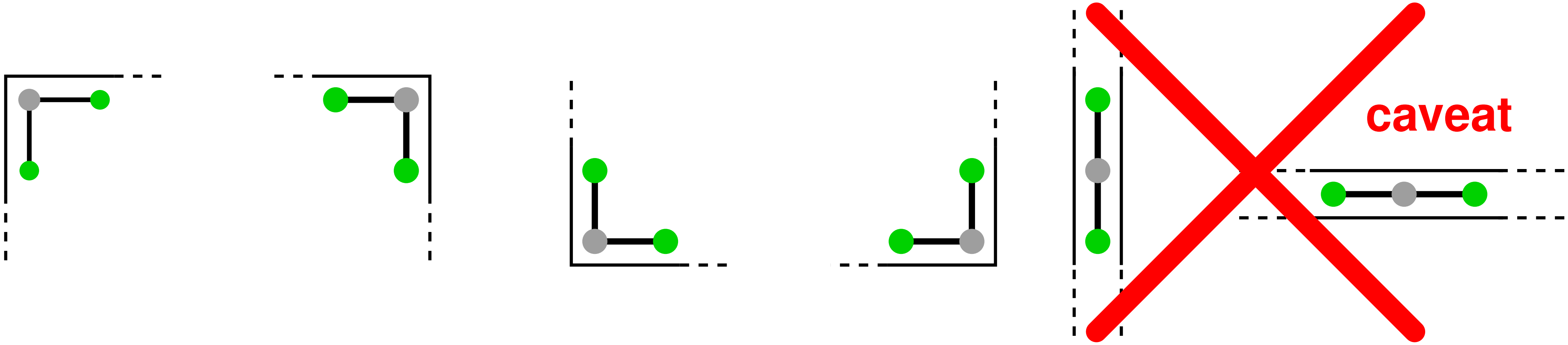
- caveat-freeness by path constraint

Example: 3-Avoiding

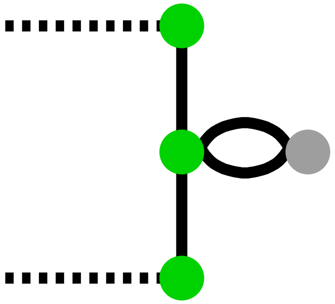
- psinglets: HPH-subsequence



- in cubic lattice: has strong influence on core

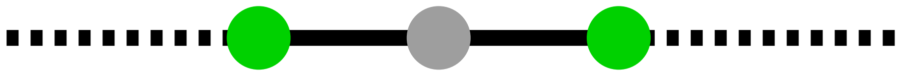


- caveat-freeness by path constraint
- remaining invalid case

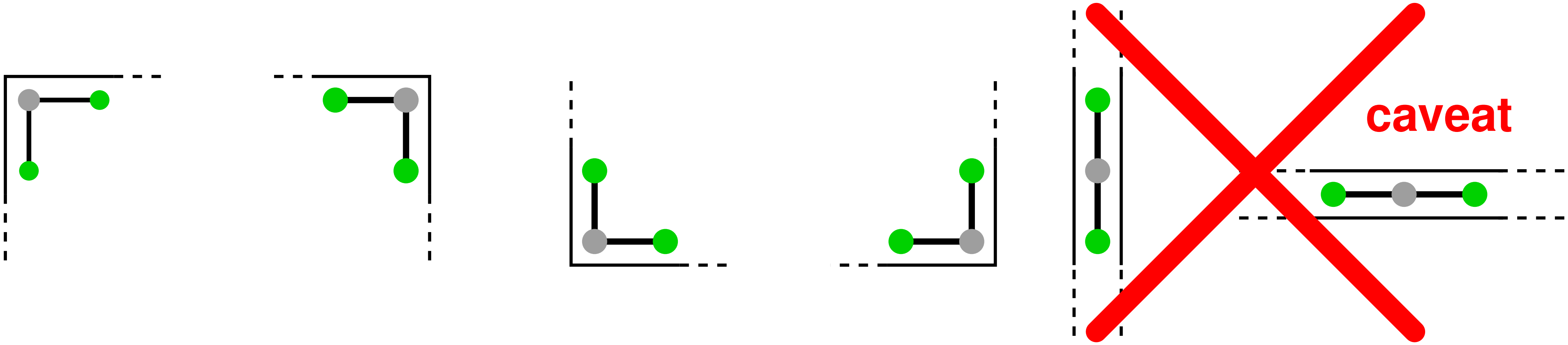


Example: 3-Avoiding

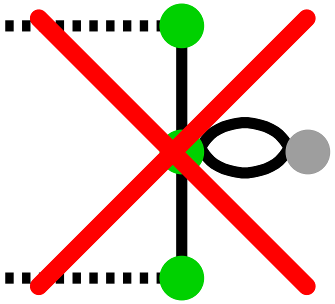
- psinglets: HPH-subsequence



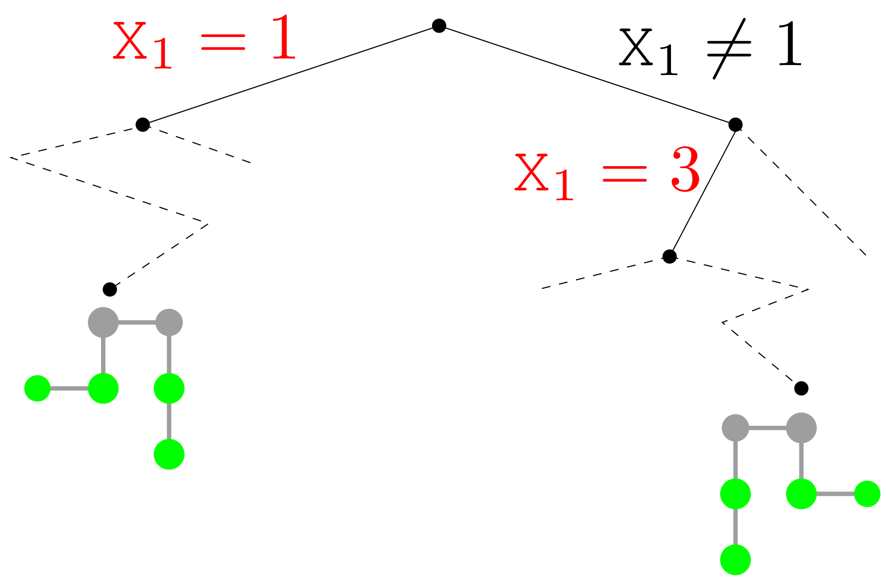
- in cubic lattice: has strong influence on core



- 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:

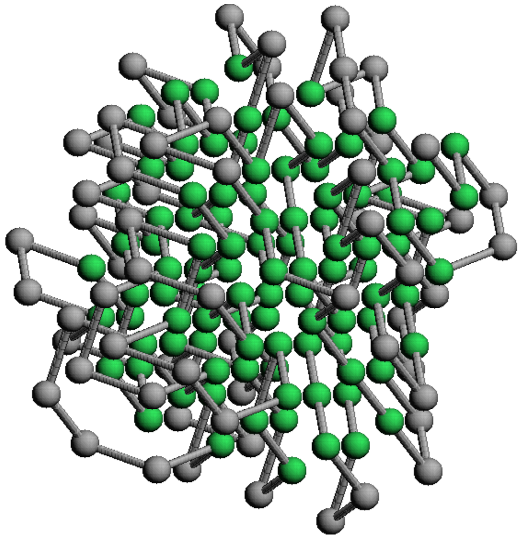
authors	model	dim.	maxlen	algorithm	comment
[Yue& Dill PhysRevE93]	cubic HP	3	36	branch-and-bound	optimality proven
[Yue&Dill PNAS95]	cubic HP	3	88	branch-and-bound	optimality proven
[Sazhin et al. 01]	cubic HP, FCC	3	34	branch-and-bound	not always optimal
[Cui et al. PNAS02]	square HP	2	18	compl. enum	
[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

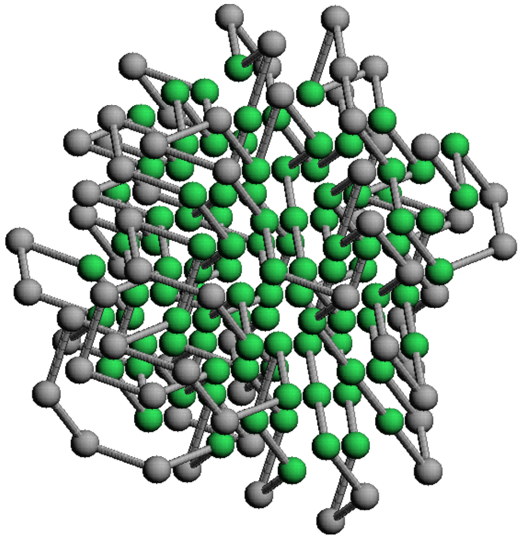
seq.	length	runtime
S1	135	9 s
S2	151	15 s
S3	161	18 s
S4	164	11 s

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**

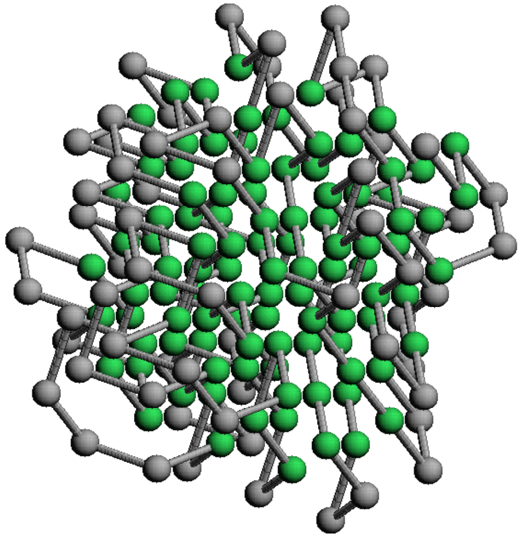
Comparison of Results

- 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	$\mathrm{HPH_2P_2H_4PH_3P_2H_2P_2HPH_3PHPH_2P_2H_2P_3HP_8H_2}$	0,1 s	6,9 min
2	$\mathrm{H_4PH_2PH_5P_2HP_2H_2P_2HP_6HP_2HP_3HP_2H_2P_2H_3PH}$	0,1 s	40,5 min
3	$\mathrm{PHPH_2PH_6P_2HPHP_2HPH_2PHPHP_3HP_2H_2P_2H_2P_2HPHP_2HP}$	4,5 s	100,2 min
4	$\mathrm{P_2HP_3HPH_4P_2H_4PH_2PH_3P_2HPHPHP_2HP_6H_2PH_2PH}$	1,8 s	74,7 min
5	$\mathrm{H_3P_3H_2PHPH_2PH_2PH_2PHP_7HPHP_2HP_3HP_2H_6PH}$	1,7 s	59,2 min
6	$\mathrm{PHP_4HPH_3PHPH_4PH_2PH_2P_3HPHP_3H_3P_2H_2P_2H_2P_3H}$	12,1 s	144,7 min
7	$\mathrm{PHPH_2P_2HPH_3P_2H_2PH_2P_3H_5P_2HPH_2PHPHP_4HP_2HPHP}$	7,3 s	284,0 min
8	$\mathrm{PH_2PH_3PH_4P_2H_3P_6HPH_2P_2H_2PHP_3H_2PHPHPH_2P_3}$	1,5 s	26,6 min
9	$\mathrm{PHPHP_4HPHPHP_2HPH_6P_2H_3PHP_2HPH_2P_2HPH_3P_4H}$	0,3 s	1420,0 min
10	$\mathrm{PH_2P_6H_2P_3H_3PHP_2HPH_2P_2HP_2HP_2H_2P_2H_7P_2H_2}$	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 algorithms

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

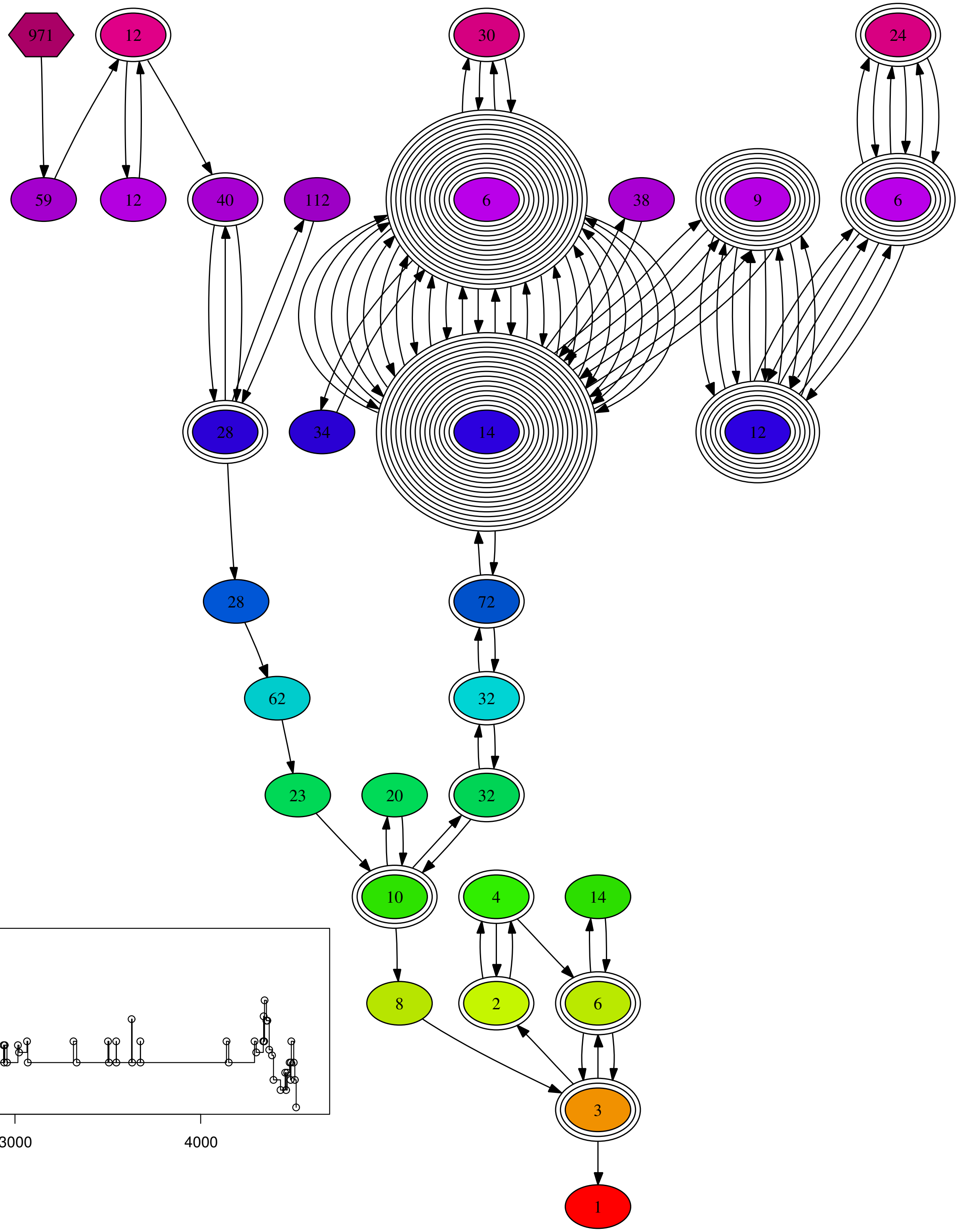
- find sequences with *exactly* *one* optimal structure
- stochastic local search

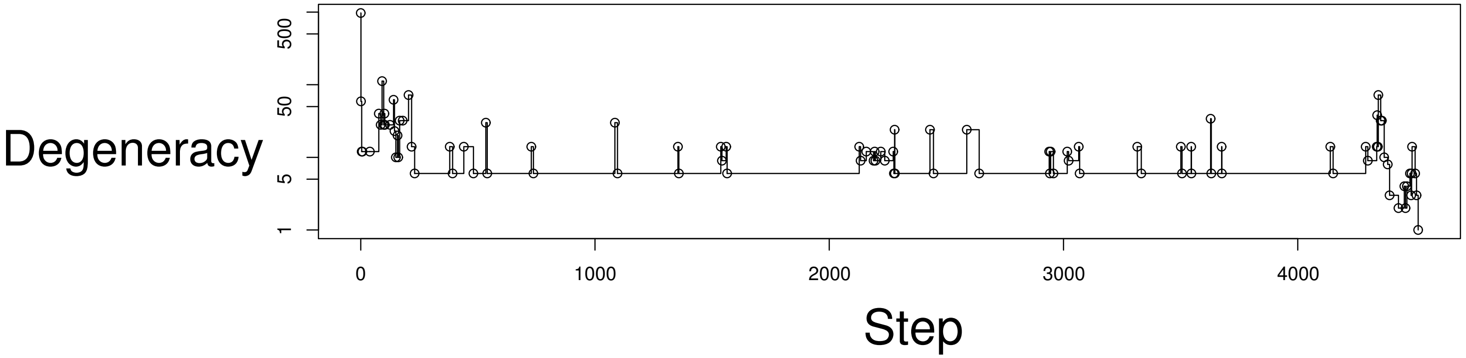
node:

accepted sequences

edges:

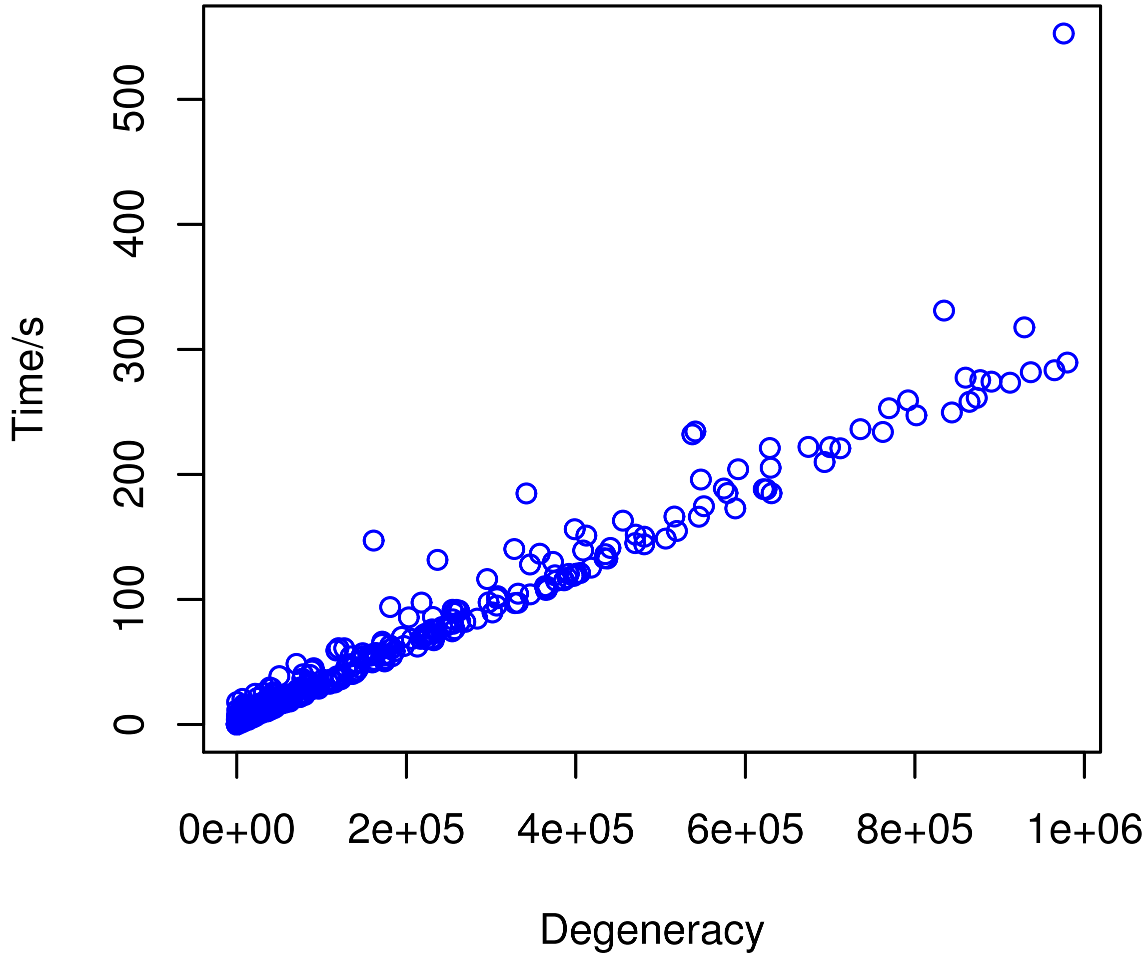
simulation step/mutation





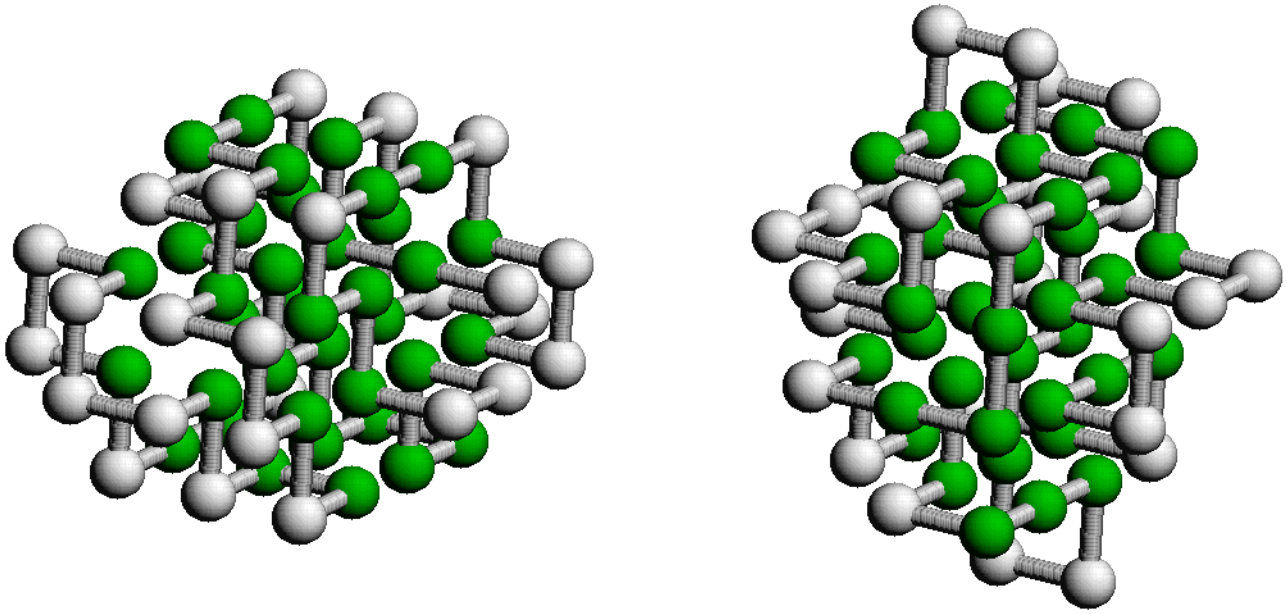
Run Time Requirements

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

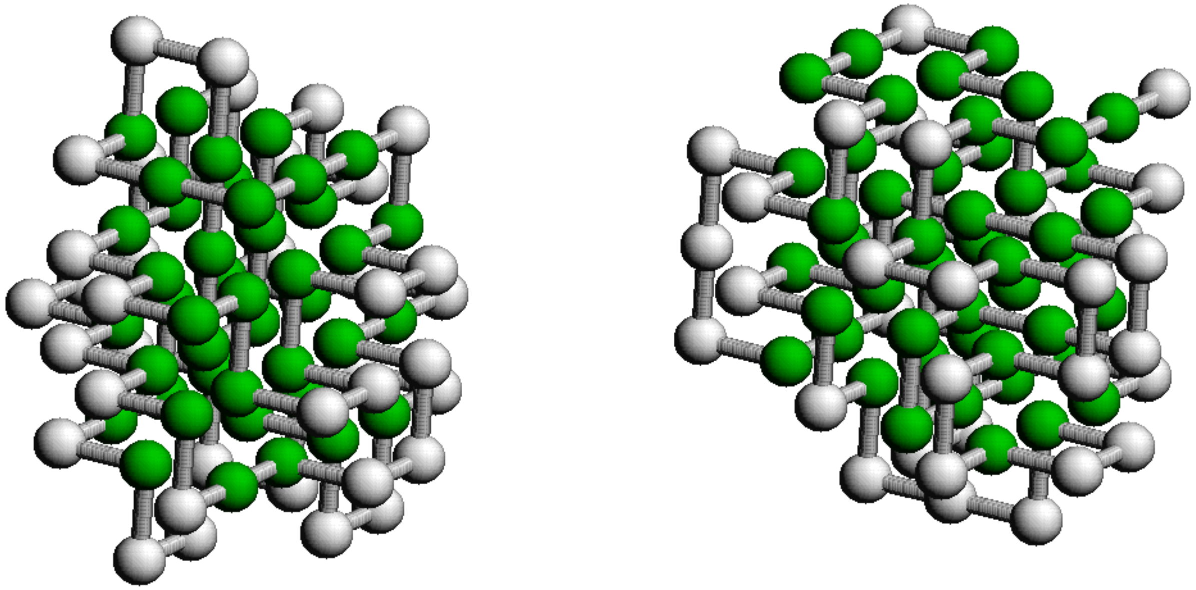


Example: Sequences with Unique Ground-State

- length 64:

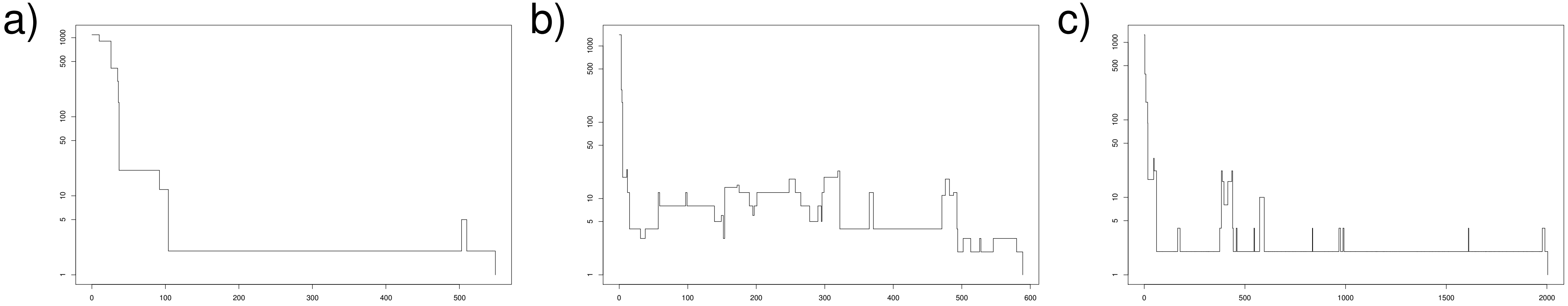


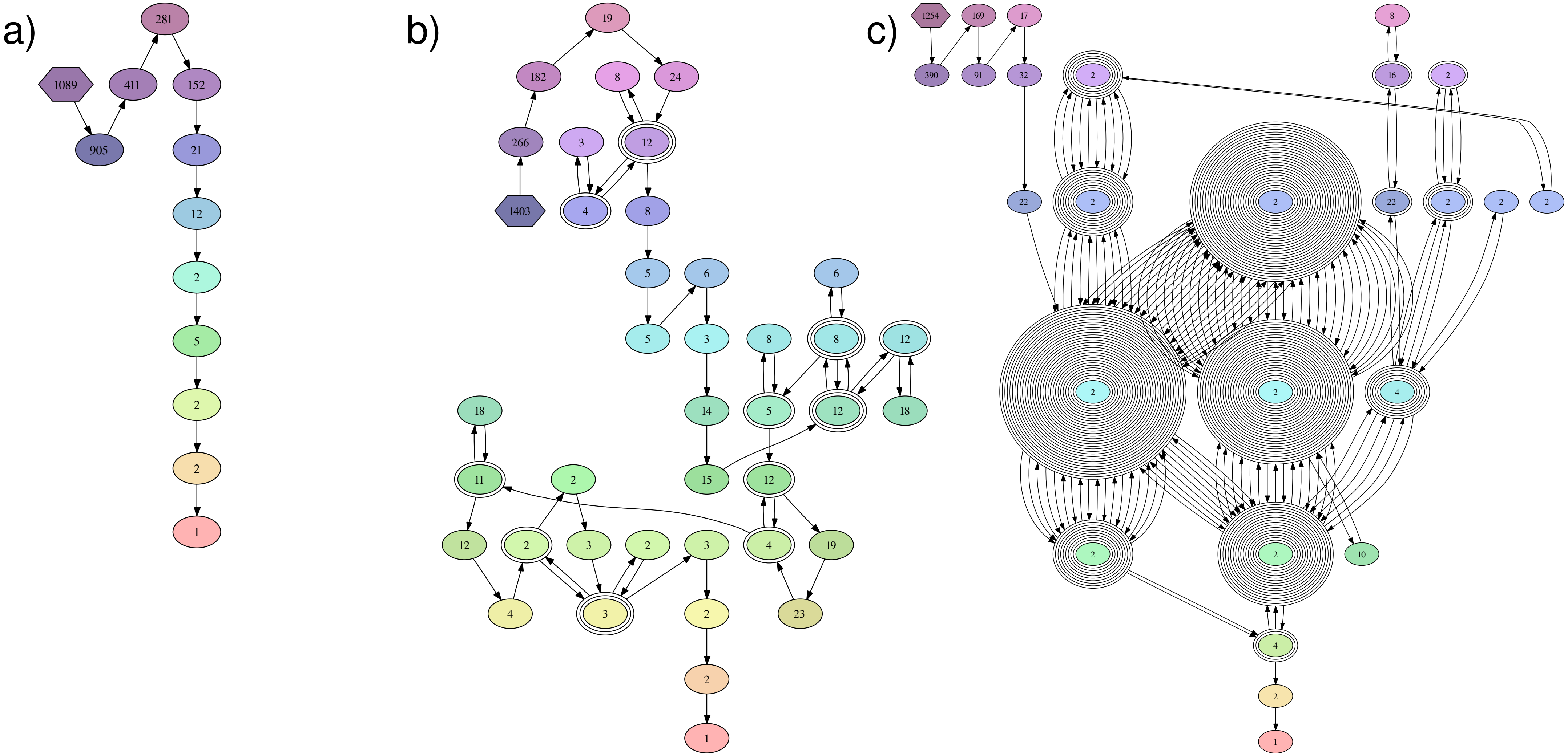
- length 80:



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

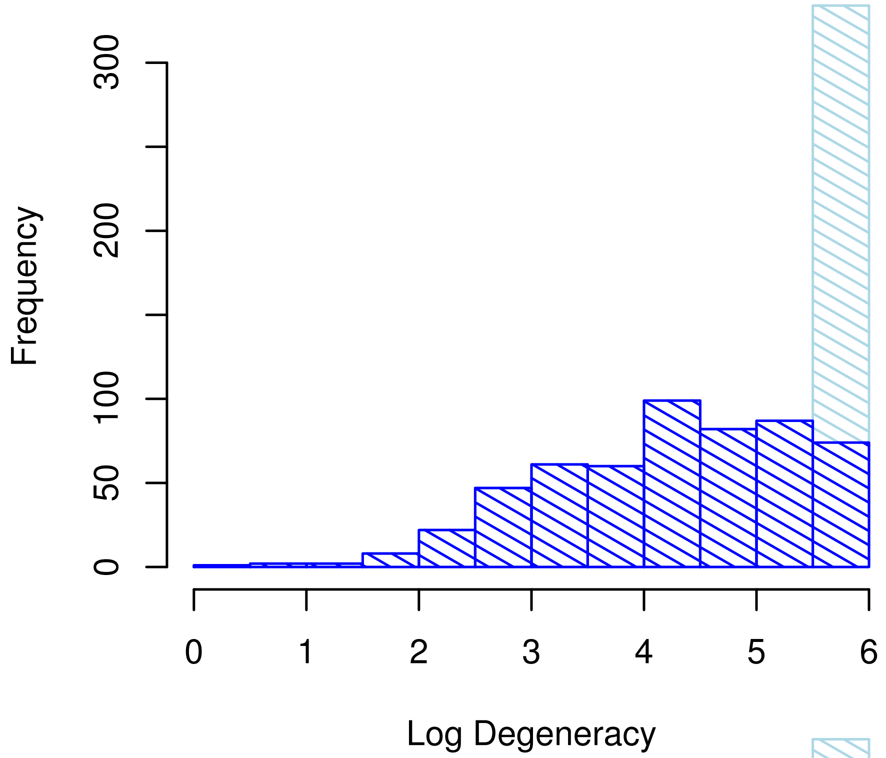
Three “Typical” Runs



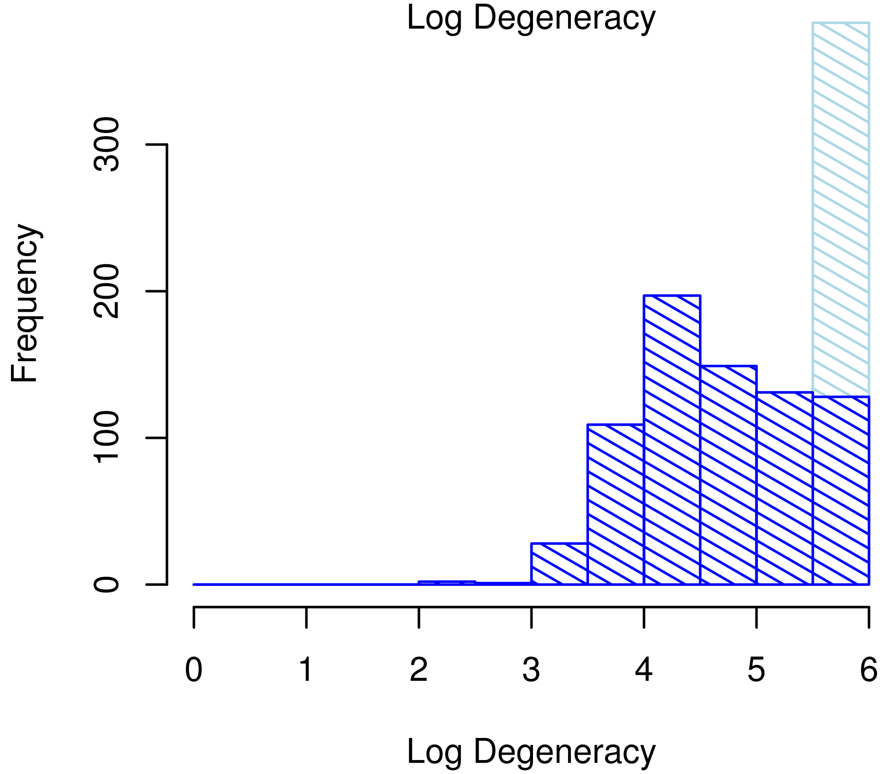


Degeneracy: FCC vs. Cubic

- log-degeneracy cubic HP-model:



- log-degeneracy FCC HP-model:



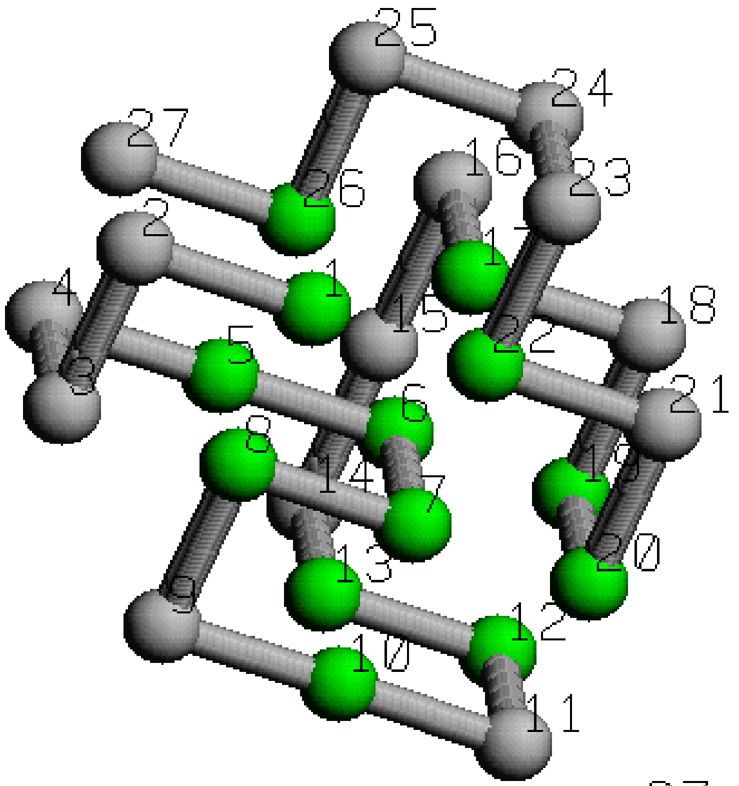
Degeneracy: HP-Model vs. HPNX-model

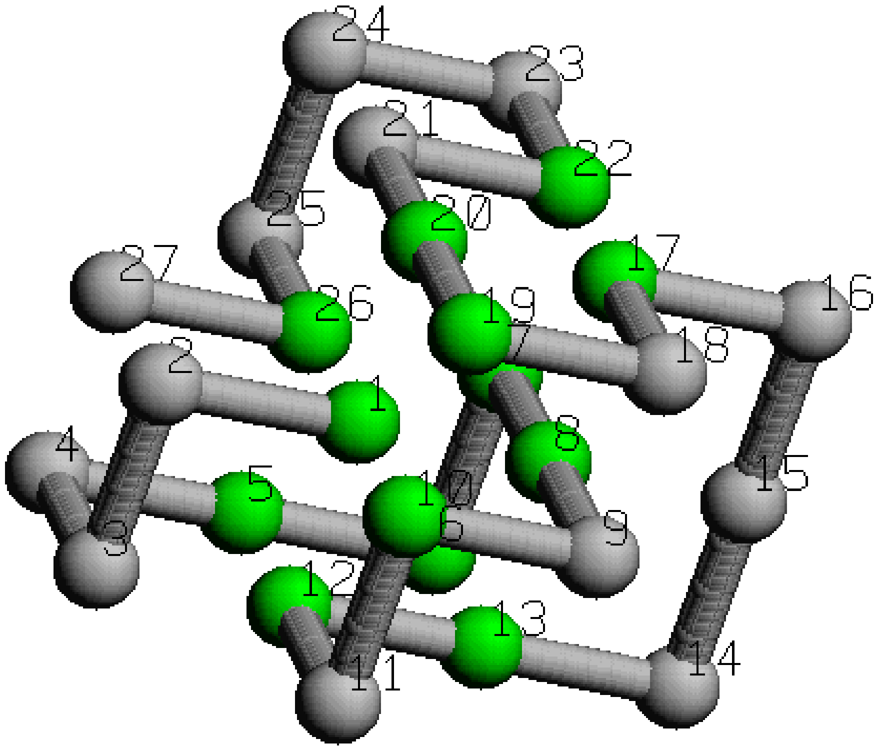
- HPNX: P=positive N=negative X=neutral
- should reduce the degeneracy

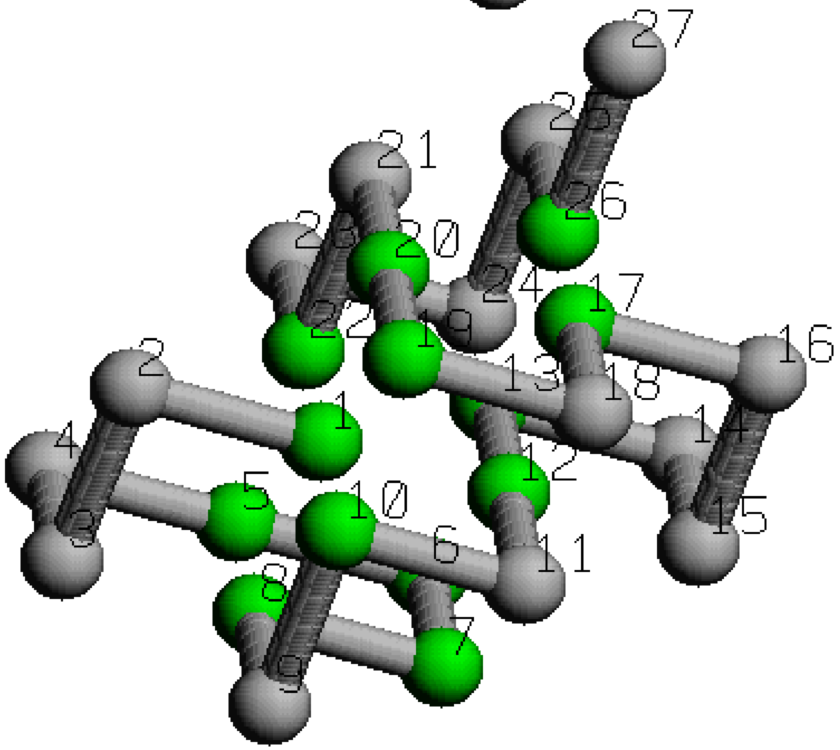
- How much? ⇒ preliminary results
  - HP: approx. 0.016% of all random sequences are uniquely folding.
  - HPNX: approx. 2.6% of all random sequences are uniquely folding.
- *Note: 50% H monomers*

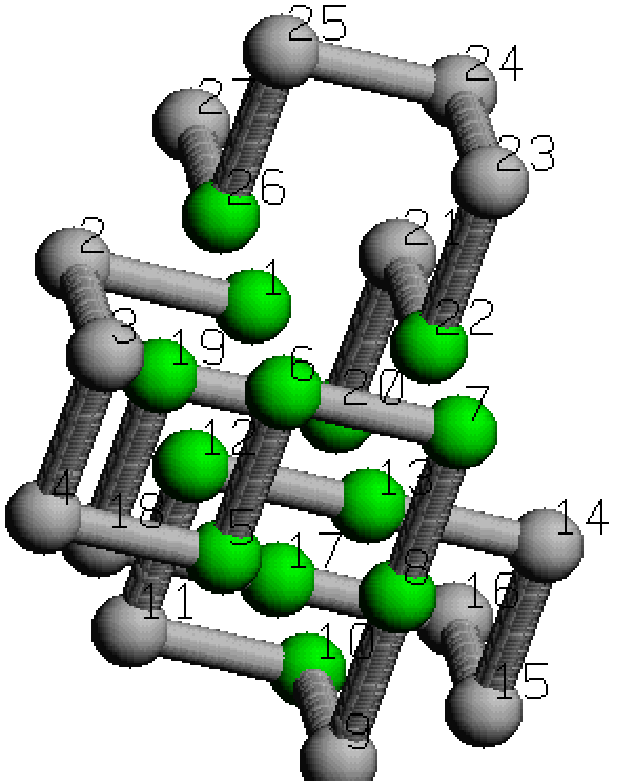
- example for reduction: sequence S2
  - HPNX: HXNNHHHHXHXHHNXNHXHHNHPPXHP
  - corresp. HP: HPPPHHHHPHPHHPPPHPHHPHPPPHP

S_2 HP-sequence: 4 out of 297

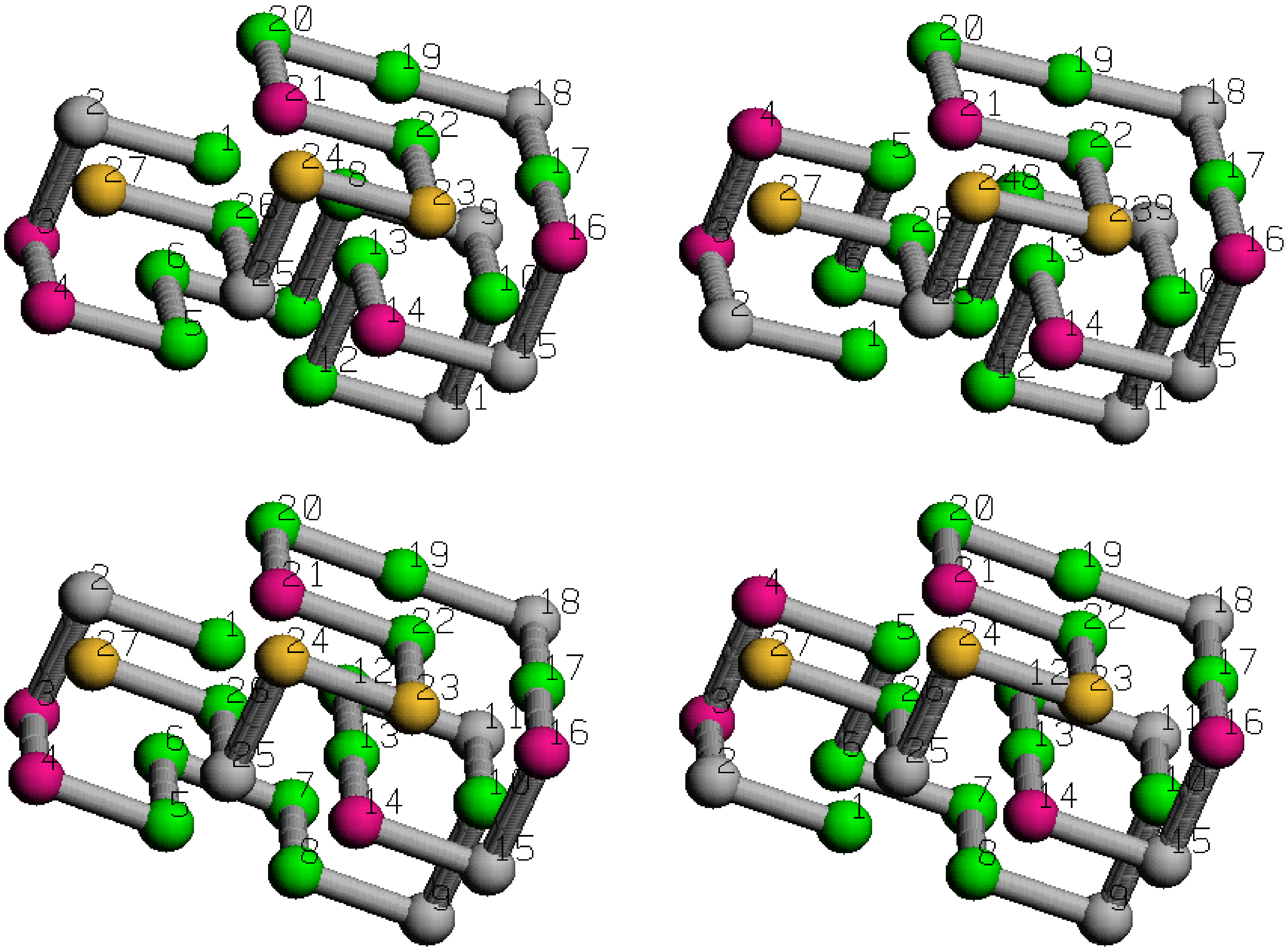




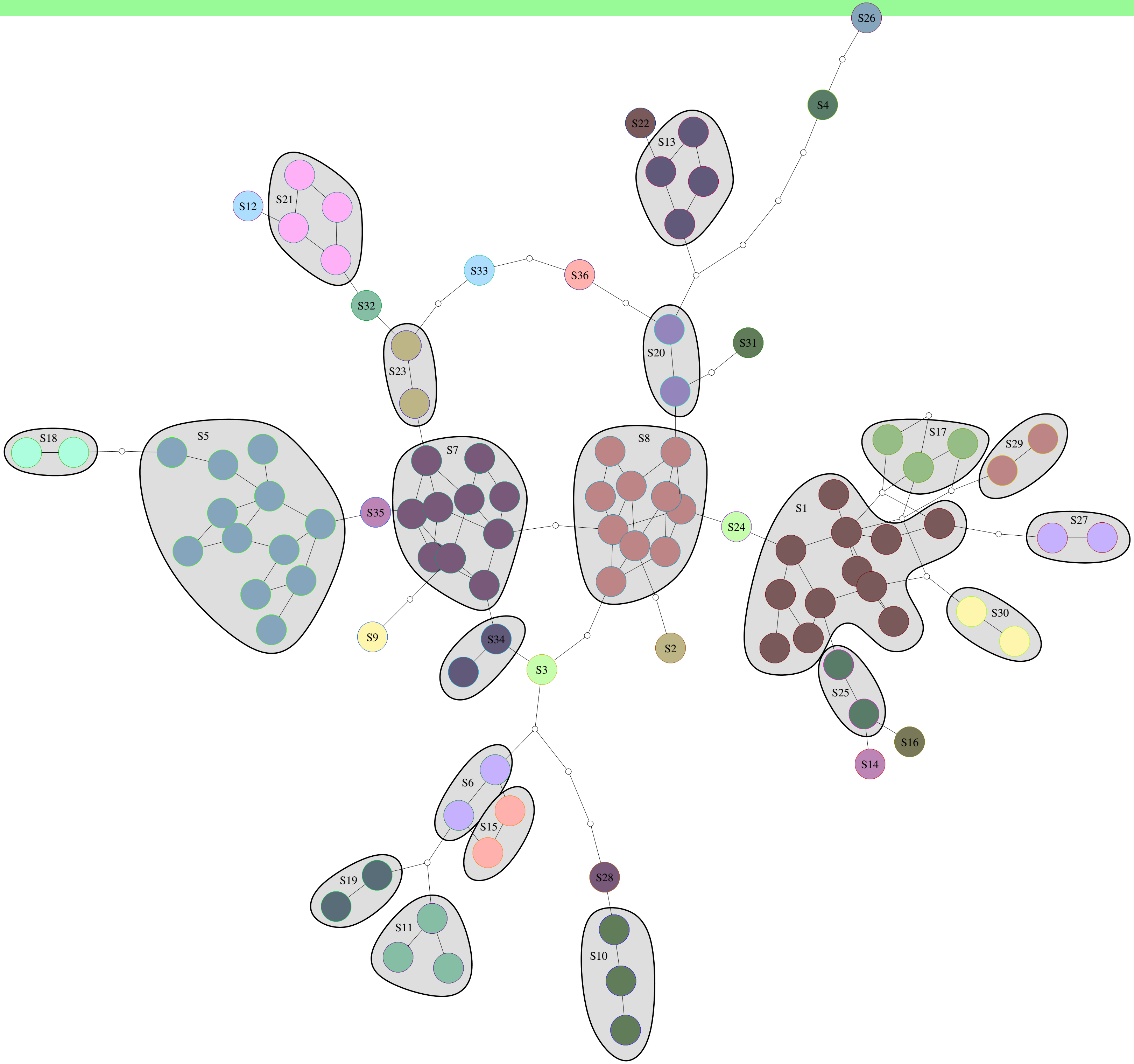




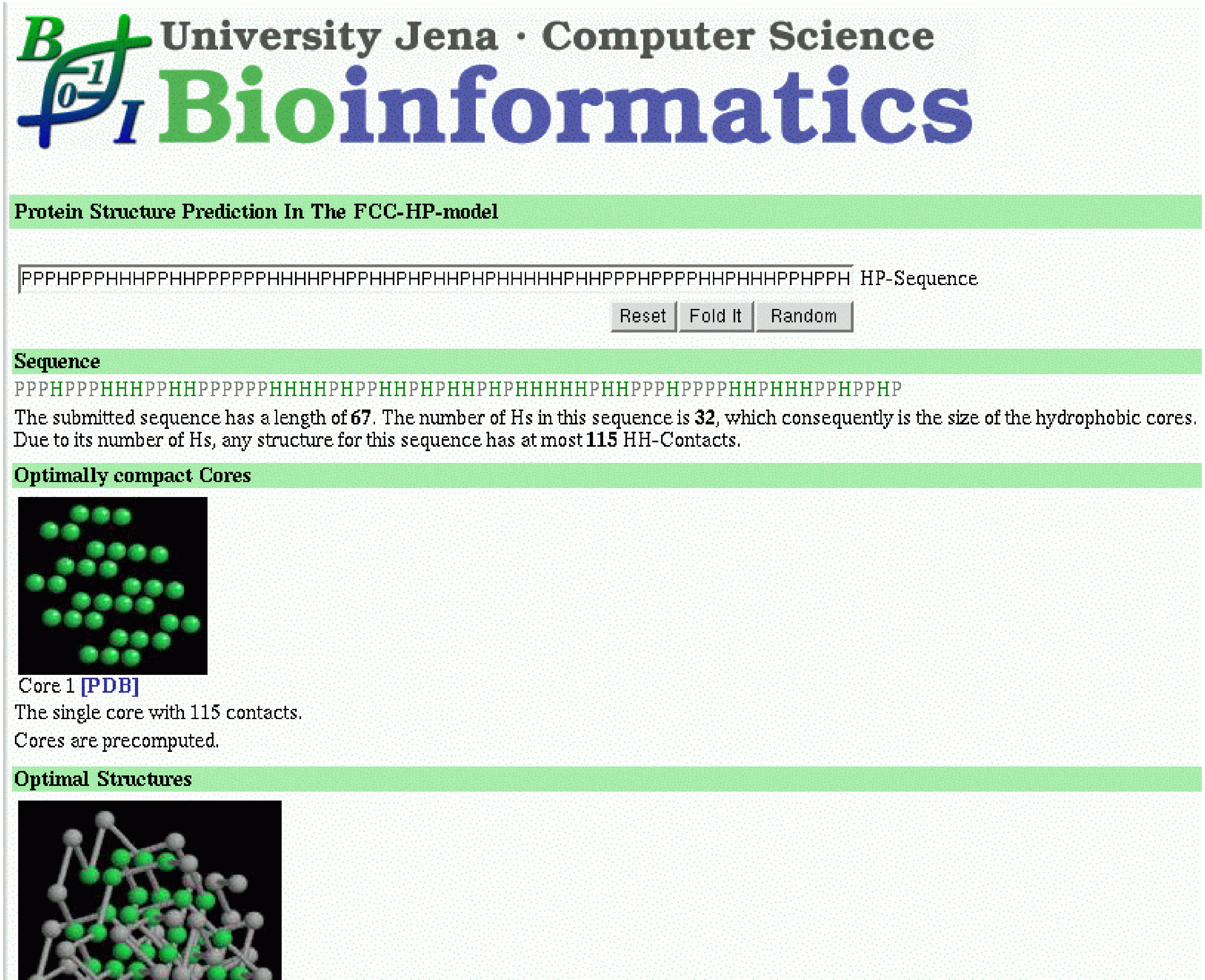
S_2 HPNX-sequence: the 4 native ones



Connectivity of Neutral Nets



WWW-Page



University Jena · Computer Science

Bioinformatics

Protein Structure Prediction In The FCC-HP-model

PPPHPPPHHHPPHHPPPPPPHHHHPHPPHHPHPHHPHPHHHHHPHHPPPHPPPPHHPHHHPPHPPH HP-Sequence

Sequence

PPPHPPPHHHPPHHPPPPPPHHHHPHPPHHPHPHHPHPHHHHHPHHPPPHPPPPHHPHHHPPHPPHP

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