



HAL
open science

Variable Neighborhood Search with Cost Function Networks to Solve Large Computational Protein Design Problems

David Allouche, Antoine Charpentier, Thomas Schiex, Thomas Simonson

► **To cite this version:**

David Allouche, Antoine Charpentier, Thomas Schiex, Thomas Simonson. Variable Neighborhood Search with Cost Function Networks to Solve Large Computational Protein Design Problems: we explored various mutator for improving VNS robustness for the protein design . But the method is generic and can be used very likely in various context of molecular modelling if energy table can be precomputed. Rencontre des Chimistes Théoriciens Francophones, Oct 2018, Toulouse, France. 2018. hal-02733727

HAL Id: hal-02733727

<https://hal.inrae.fr/hal-02733727>

Submitted on 2 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Variable Neighborhood Search with Cost Function Networks to Solve Large Computational Protein Design Problems

Contact : David.allouche@inra.fr

A. Charpentier^{1,2,4}, D. Mignon³, S. Barbe¹, J. Cortes², T. Schiex⁴, T. Simonson³, D. Allouche⁴

download : <http://github.com/toulbar2>

¹ LISBP (INSA Toulouse), ² LAAS-CNRS Toulouse, ³ Lab biochimie Ecole Polytechnique Palaiseau, ⁴ MIAT UR 875, INRA Castanet Tolosan

Summary

Graphical models factorize a global probability distribution/energy function as the product/sum of local functions. A major inference task, known as MAP in Markov Random Fields and MPE in Bayesian Networks, is to find a global assignment of all the variables with maximum a posteriori probability/minimum energy. A usual distinction on MAP solving methods is complete/incomplete, i.e. the ability to prove optimality or not. Most complete methods rely on tree search, while incomplete methods rely on local search. Among them, we study Variable Neighborhood Search for graphical models. **In this work, first we explored various mutator for improving VNS robustness for the protein design. But the method is generic and can be used very likely in various context of molecular modelling if energy table can be precomputed.**

Cost Function Networks ↔ Markov Random Field

(Shapiro, Haralick, IEEE PAMI 81)

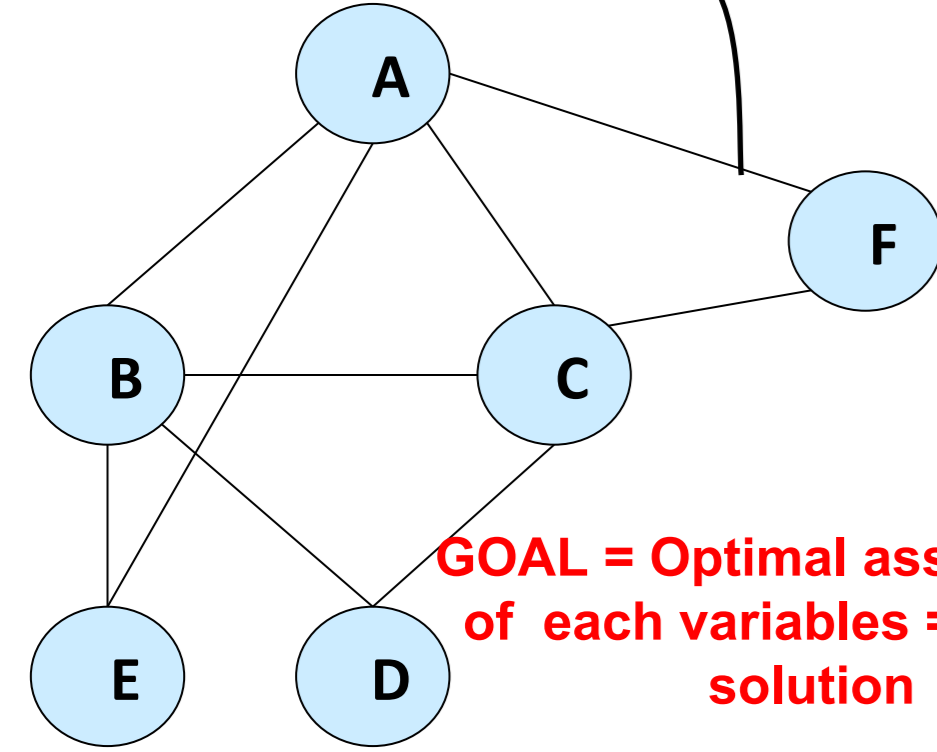
- n variables
 - finite domains
- e local/global functions
 - scope, function with costs
- Costs $\in \{0, \dots, +\infty\}$ finite or infinite integer

$$X = \{X_1, \dots, X_n\}$$

$$x_i \in D_i, |D_i| \leq d$$

$$F = \{f_1, \dots, f_e\}$$

Minimize $\sum_F f_i(X)$ NP-hard



GOAL = Optimal assignment of each variables = optimal solution

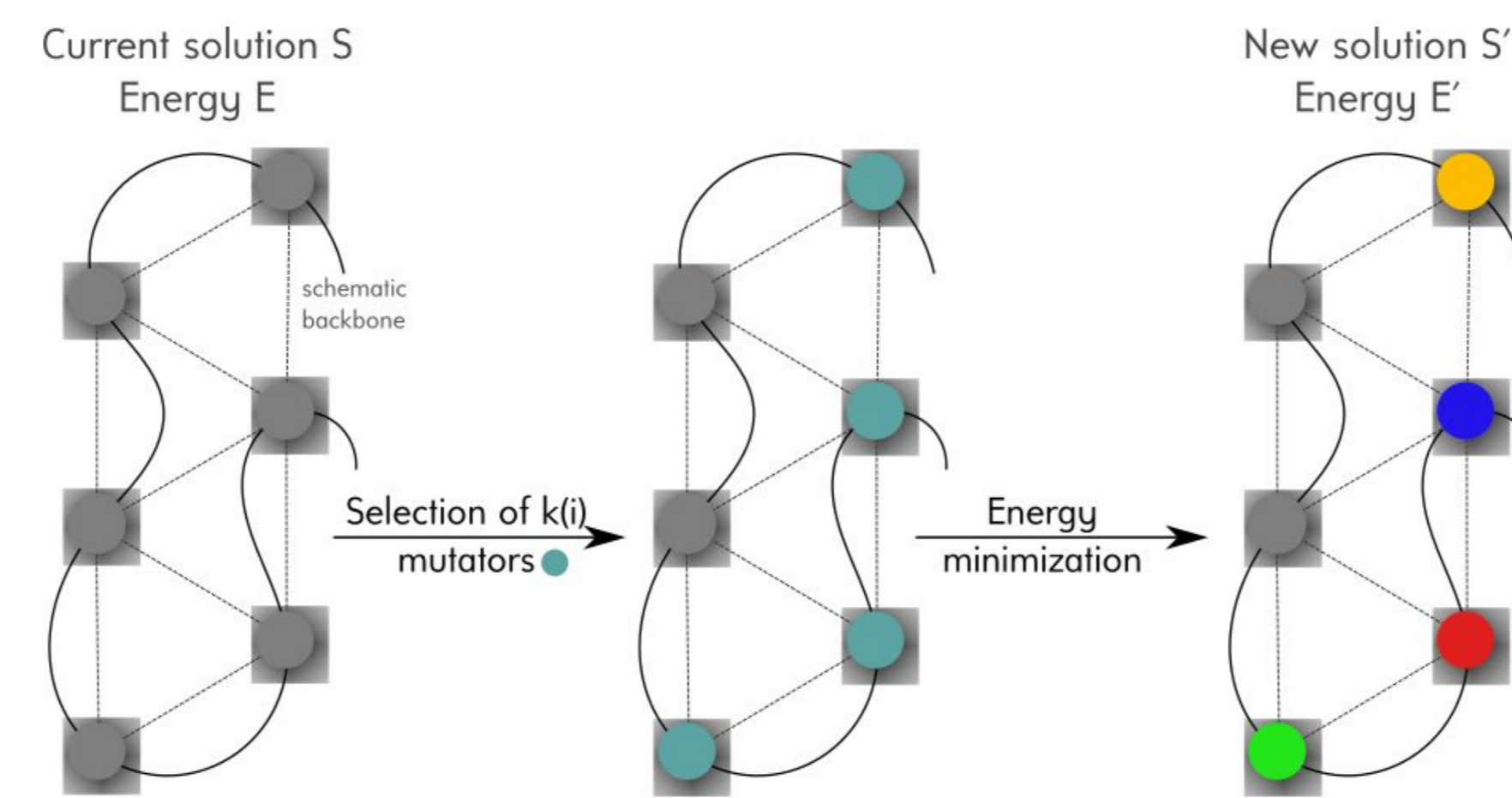
$$\mathbb{P}(x) = \frac{1}{Z} \prod_{f_S \in F} f_S(x_S)$$

$$Z = \sum_{x \in \Delta} \prod_{f_S \in F} f_S(x_S)$$

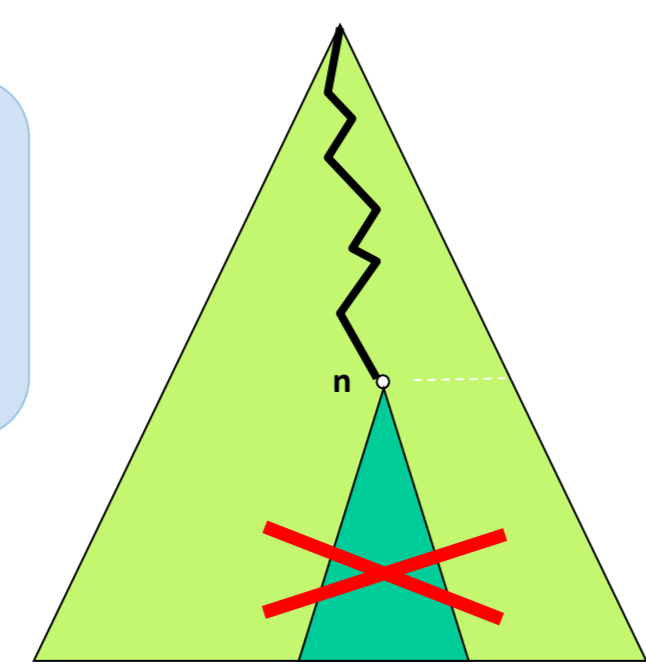
For optimization (MAP = Maximum a posteriori), MRF and CFN are essentially equivalent after a $-\log$ transform.

GMEC = MAP = Global OPTIMAL solution = solution with minimal Energy

Variable Neighbourhood Search

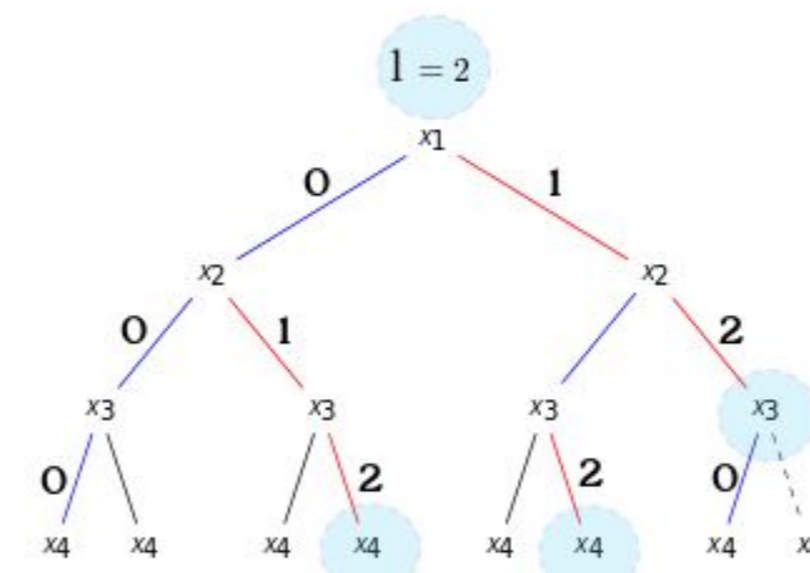


Branch-and-Bound Search



Upper Bound UB
Lower Bound LB(n)
Prune if $LB(n) \geq UB$
estimates the optimal cost below n

Limited discrepancy search



LUBY Mutator (growing slow down)

Optimal speedup of Las Vegas algorithms = **Luby serie**

Luby strategy (supposed to be optimal when not particular knowledge about a given random algorithm)

Let i the current iteration of the algorithm (i.e. search on y current neighborhood)

Let $a = \ln(i+1)$ and $b = 2^{a-1}$;

$$luby(i) = \begin{cases} b, & \text{if } i = b \\ s_{univ} = (1,1,2,1,1,2,4,1,1,2,1,1,2,4,8,\dots) \\ luby(b-1) \end{cases}$$

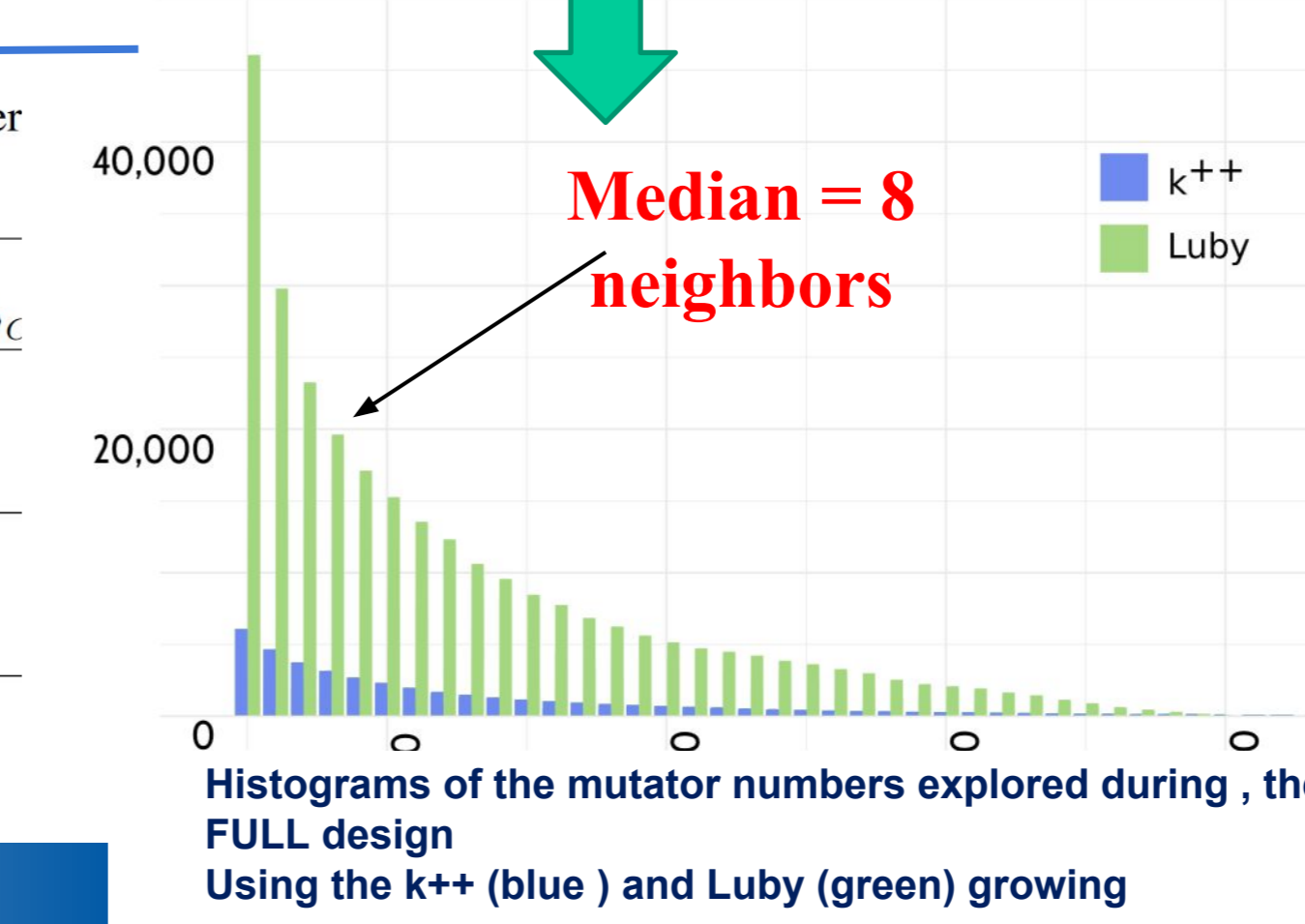
If $luby(i) = 2^m$, $k(i) = k(i-1) + 1$
Otherwise, $k(i) = k(i-1)$

if ($m=0$) sequence of increments becomes

$\{1,1,0,1,1,0,0,1,1,0,1,1,0,0,0,1,\dots\}$

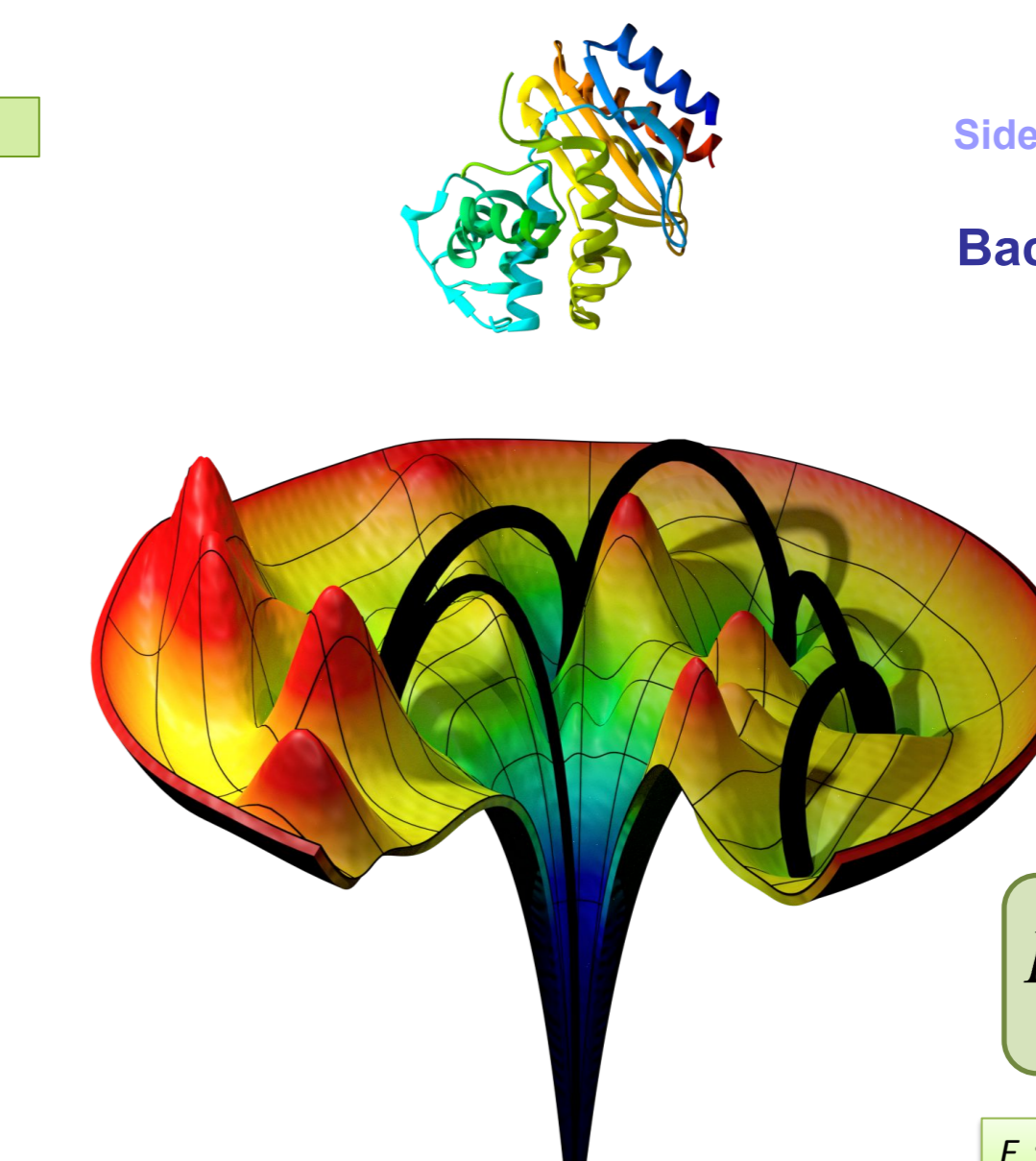
with a larger m , the mutator number increases even more slowly

With $m=3 \Rightarrow 2^3=8$

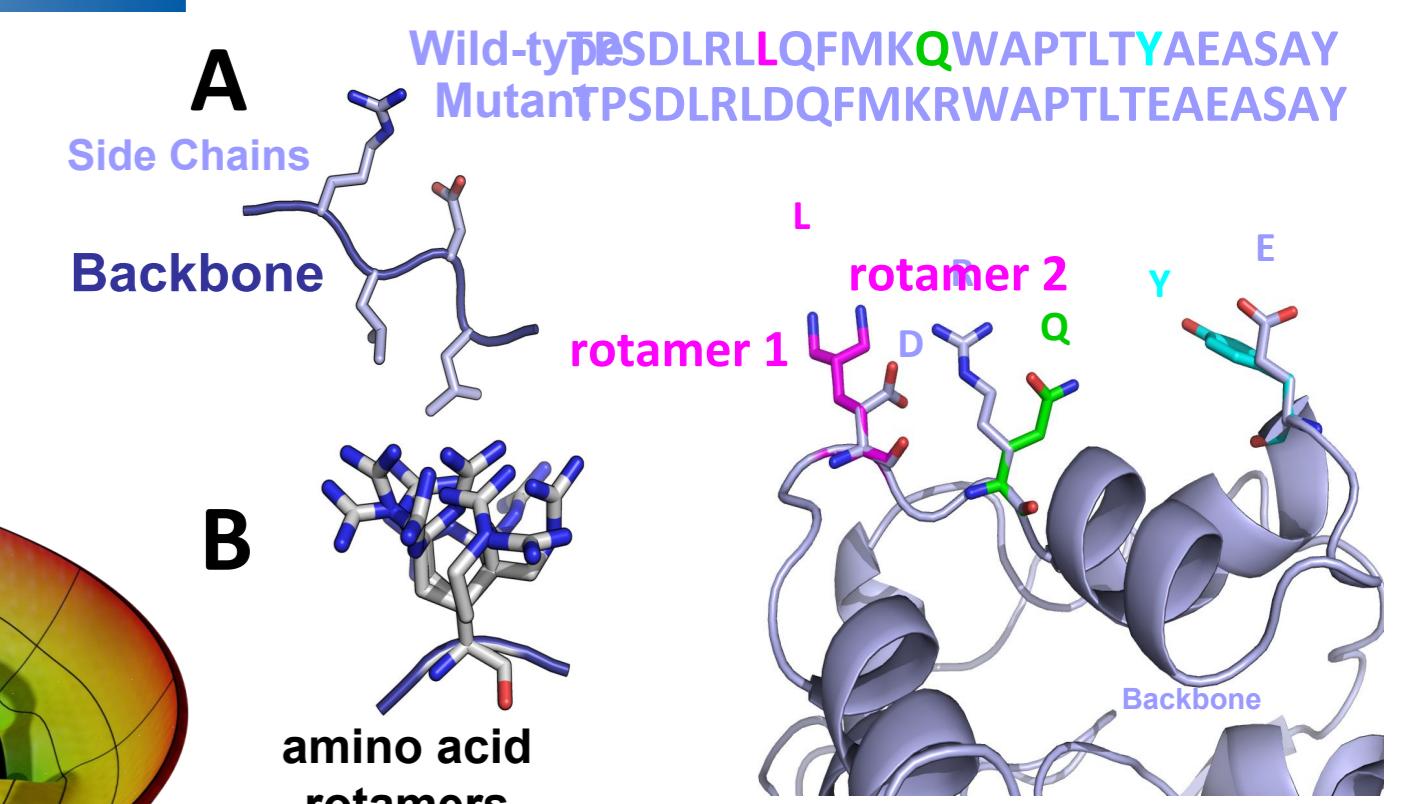


CPD FORMULATION

Mutation landscape



Amino acid sequences



$$E = E_T + \sum_i E(i_r) + \sum_{i,j>i} E(i_r, j_s)$$

E_T : self backbone energy
 $E(i_r)$: backbone-rotamer energy
 $E(i_r, j_s)$: rotamer-rotamer energy

COMPUTATIONAL PROTEIN DESIGN

Precomputation and storage of all position

Designed positions

Energy Table

	A1	V1	V2	V3	S1	S2	S3
A1	0.3	-0.1	-1.2	-1.1	-0.9	4.1	2.3
V1	-2.7	-0.3	2.3	4.7	8.1	1.9	8.5
V2	5.6	-0.2	3.1	5.2	6.3	1.3	4.5
V3	8.2	10.6	2.9	6.4	5.8	0.9	3.2
S1	-1.9	1.7	4.3	8.2	4.4	-0.9	-2.6
S2	7.9	2.1	5.7	6.4	8.1	-0.8	-4.9
S3	-0.4	3.7	6.4	2.9	7.7	-0.4	-5.5

Pairwise interaction

COST Function Network

Allowed rotamers

P1 P2

A1 A1
V1 V1
V2 V2
V3 V3
S1 S1
S2 S2
S3 S3

Benchmarking Problem set (mignon & al jcc 2015)

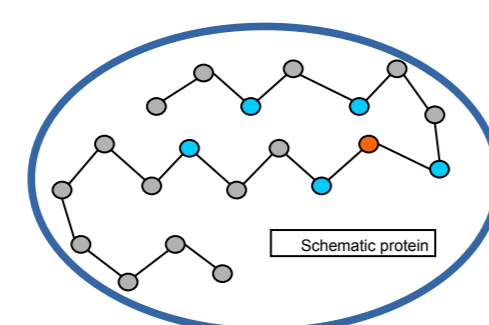
PDB	# MUT
SH3 1ABO	48
SH3 1CKA	49
PDZ 1R6J	72
PDZ 1G9O	76
PDZ 2BYG	82
SH2 1BM2	88
SH2 1M61	88
SH2 1A81	92
SH2 1O4C	96

- Search space size (248#mut):
 - N20: $\square [2 \cdot 10^{61}, \dots, 9 \cdot 10^{123}]$
 - N30: $\square [1 \cdot 10^{82}, \dots, 2 \cdot 10^{137}]$
 - FULL: $\square [8 \cdot 10^{114}, \dots, 7 \cdot 10^{229}]$

- 99 problems
- AMBER Force field
- CASA implicit solvent Model
- Tuffery rotamer library (248 rotamers)
- Non mutated positions remains flexible (except GLY, PRO position)

VNS	Full		N30		N20	
	+	0	+	0	+	0
Luby/d-prob	98%	0%	65%	31%	31%	68%
Luby/random	99%	0%	63%	31%	31%	69%
Luby/nearest	94%	0%	60%	31%	28%	66%
k++/d-prob	80%	0%	56%	27%	27%	66%
k++/nearest	80%	0%	54%	21%	21%	62%

d prob selection



$$p_{i,j} = \frac{1}{d_{i,j} * \sum_{k=1, k \neq i}^d d_{i,k}}, i \neq j$$

$$p_{i,i} = 0$$

Table 3: Averages $\langle \Delta E \rangle_C$ of the mean energy improvement, compared to MC+REMC+SDH, over the test category C, and average $\langle \sigma(\Delta E) \rangle_C$ of its standard deviation.

	Full	N30	N20
VNS	3.17	1.03	0.77
Luby/d-prob	2.95	1.06	0.76
Luby/random	2.78	1.36	0.69
Luby/nearest	0.77	2.40	0.59
k++/d-prob	1.03	2.21	0.43
k++/nearest	0.55	2.47	0.58

FULL DESIGN Energy Gap Box plot

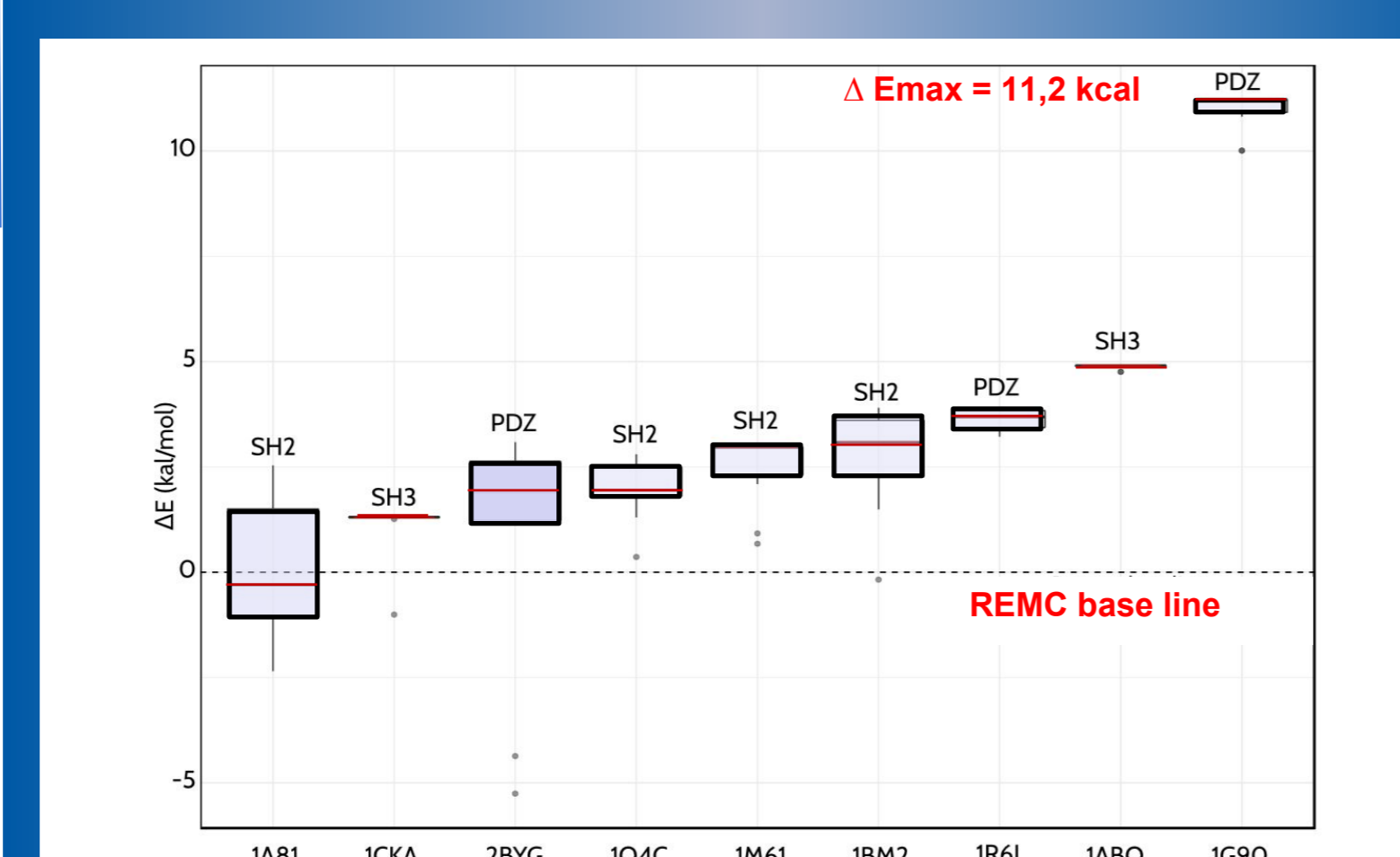


Figure 5: Box plot of ΔE values with VNS Luby/d-prob from ten runs on each Full test.

Optimal solution recovery rate

heuristic	Full (90)	%	N30 (450)	%	N20 (450)	%
k++/d-prob	10	0.111	320	0.711	425	0.944
k++/k-nearest	10	0.111	258	0.573	405	0.90
k++/random	15	0.167	319	0.709	440	0.978
Luby/d-prob	40	0.444	407	0.904	447	0.993
Luby/k-nearest	33	0.367	356	0.791	438	0.973
Luby/random	37	0.411	401	0.891	449	0.998

For each Heuristic (line) and type of instance (columns) the table give the number and the recovery rate of the optimal Solution over the 90, 450 runs respectively for the FULL, N20 and N30 set of problems.

COMPUTATION TIME LESS THAN 2H on 1CPU
Parallel release available

IMPORTANT MESSAGES

- VNS converge faster than the other global search methods.
- Luby + probabilistic selection based on variable distance improves VNS robustness
- VNS allows to explore large search space
- VNS + cost function is compliant with n-body energetic terms.
- polarisable force field or fragment based method with QM description could be used for improving the model.