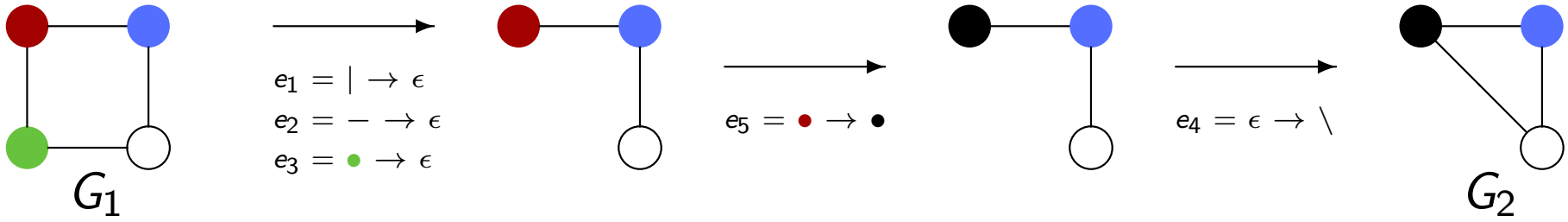
Distance d'édition entre graphes vue comme la minimisation d'une fonctionnelle quadratique

E. Daller, N. Boria, S. Bougleux, B. Gaüzère and L. Brun

Normandie Univ
Greyc – Université de Caen Normandie

Agenda

Graph Edit Distance



Example of edit sequence $\gamma \in \Gamma(G_1, G_2)$

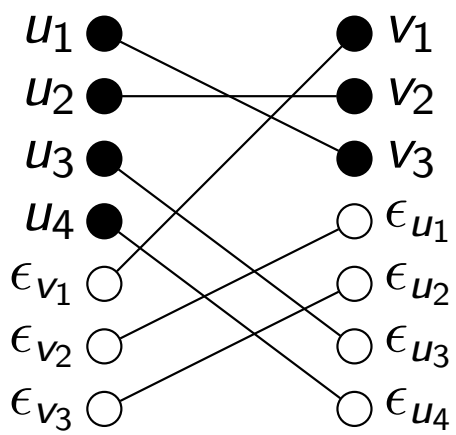
Edit costs

Each edit operation e_k is penalized by a cost $c(e_k)$

$$\begin{aligned} \mathrm{GED}(G_1, G_2) &= \min_{\gamma \in \Gamma(G_1, G_2)} \left\{ \sum_{e \in \gamma} c(e) \right\} && (1) \\ &= \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^\top \Delta \mathbf{x} + \mathbf{c}^\top \mathbf{x} \right\} && (2) \\ &= \min_{\mathbf{x}} Q(\mathbf{x}) && (3) \end{aligned}$$

GED as a Quadratic Assignment Problem

Find an assignment $\mathbf{x}$ between the nodes of the two graphs under the following constraint :



$$\mathrm{GED}(G_1, G_2) = \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^\top \Delta \mathbf{x} + \mathbf{c}^\top \mathbf{x} \right\}$$

with cost matrices : Δ for edge assignement and $\mathbf{c}$ for node assingment

Complexity

QAP, and thus GED computation is NP-hard.

Estimate Graph Edit Distance

- Bipartite GED

  [Riesen, Neuhaus, Bunke. GbR 2007]
  [Riesen, Bunke. Image and Vision Computing 2009]
  [Gaüzère et al. SSPR 2014]
  [Carletti et al. GbRPR 2015]
  [Serratosa. Image and Vision Computing 2015]

- Greedy bipartite GED

  [Riesen et al. GbRPR 2015]
  [Fisher et al. Pattern Recognition Letters 2017]

- Refinements methods (*e.g.* Beam search)

  [Riesen, Bunke. Pattern Recognition 2015]
  [Riesen. Advances in Computer Vision and Pattern Recognition 2015]

- Simulated annealing

  [Riesen et al. GbRPR 2017]

- Frank-Wolfe

  [Bougleux et al. Pattern Recognition Letters 2017]

- GNCCP

  [Gaüzère et al. SSPR 2016]

Agenda

Bipartite GED : A linear approximation

Approximate the QAP by a Linear Sum Assignment Problem (LSAP) :

$$\mathbf{x}_0 \in \underset{\mathbf{x}}{\operatorname{argmin}}\{\tilde{\mathbf{c}}^\top \mathbf{x}\} \qquad \text{then} \qquad \text{bGED}(G_1, G_2) = Q(\mathbf{x}_0)$$

Where $\tilde{\mathbf{c}}$ encode the cost of matching local structures such as :

- Stars [Riesen and Bunke 2009]
- Random walks [Gaüzère et al 2014]
- Sub-graphs [Carletti et al 2015] (cf. talk of A. Inokuchi)

Bipartite GED : A linear approximation

Approximate the QAP by a Linear Sum Assignment Problem (LSAP) :

$$\mathbf{x}_0 \in \underset{\mathbf{x}}{\operatorname{argmin}}\{\tilde{\mathbf{c}}^\top \mathbf{x}\} \qquad \text{then} \qquad \text{bGED}(G_1, G_2) = Q(\mathbf{x}_0)$$

Where $\tilde{\mathbf{c}}$ encode the cost of matching local structures such as :

- Stars [Riesen and Bunke 2009]
- Random walks [Gaüzère et al 2014]
- Sub-graphs [Carletti et al 2015] (cf. talk of A. Inokuchi)

Remark

- ▷ There exist several solutions $\mathbf{x}_0$
- ▷ Each one gives a different GED estimation
- ▷ In the literature : the choice of $\mathbf{x}_0$ is arbitrary

Reformulation of bipartite GED

- ▷ There exist several solutions $\mathbf{x}_0$
- ▷ Each one gives a different GED estimation

Reformulation

We can reformulate Bipartite GED with the following optimization problem :

$$\min_{\mathbf{x}_0} \quad \frac{1}{2}\mathbf{x}_0^\top \Delta \mathbf{x}_0 + \mathbf{c}^\top \mathbf{x}_0 \tag{4}$$

$$\text{s.t.} \quad \mathbf{x}_0 \in \operatorname*{argmin}_{\mathbf{x}} \tilde{\mathbf{c}}^\top \mathbf{x} \tag{5}$$

Number of bipartite optimal solutions

Dataset	Riesen and Bunke (2009)				Gaüzère et al. (2014)			
	max(K)	min(K)	$\bar{K}$	$\sigma(K)$	max(K)	min(K)	$\bar{K}$	$\sigma(K)$
Alkane	>10000	1	6198	3847	>10000	1	993	1977
Acyclic	>10000	1	2313	3496	>10000	1	182	857
MAO	>10000	48	7695	3711	>10000	1	948	2714
PAH	>10000	>10000	10000	0	>10000	128	9894	879
CMU	16	1	1.6	1.5	-			

Table: Number K of optimal bipartite solutions with 2 cost matrix $\tilde{\mathbf{c}}$, and empirical mean and median on different datasets (Chemical and Geometric)

Improve Bipartite GED

Goal : improve accuracy

Consider k solutions to the LSAP to approximate the GED

$$\text{Let } \mathcal{S}_k \subset \underset{\mathbf{x}}{\operatorname{argmin}}\{\tilde{\mathbf{c}}^\top \mathbf{x}\} \quad \text{with} \quad |\mathcal{S}_k| = k$$

Determine $\mathcal{S}$ by Uno's algorithm in time $O((m+n)k)$.[1]

$$\text{mbGED}_k(G_1, G_2) = \min_{\mathbf{x} \in \mathcal{S}_\mathbf{k}} \left\{ \frac{1}{2}\mathbf{x}^\top \Delta \mathbf{x} + \mathbf{c}^\top \mathbf{x} \right\} \tag{6}$$

[1] T. Uno. "Algorithms for Enumerating All Perfect, Maximum and Maximal Matchings in Bipartite Graphs". In: *Algorithms and Computation* vol 1350 (1997), pp. 92–101.

Improve Bipartite GED

Goal : improve accuracy

Consider k solutions to the LSAP to approximate the GED

$$\text{Let } \mathcal{S}_k \subset \underset{\mathbf{x}}{\text{argmin}}\{\tilde{\mathbf{c}}^\top \mathbf{x}\} \quad \text{with} \quad |\mathcal{S}_k| = k$$

Determine $\mathcal{S}$ by Uno's algorithm in time $O((m+n)k)$.[1]

$$\text{mbGED}_k(G_1, G_2) = \min_{\mathbf{x} \in \mathcal{S}_\mathbf{k}} \left\{ \frac{1}{2}\mathbf{x}^\top \Delta \mathbf{x} + \mathbf{c}^\top \mathbf{x} \right\} \tag{6}$$

Remark : $\mathcal{S}_k \subset \mathcal{S}_{k+1} \implies mbGED_k(G_1, G_2) \geq mbGED_{k+1}(G_1, G_2)$

[1] T. Uno. "Algorithms for Enumerating All Perfect, Maximum and Maximal Matchings in Bipartite Graphs". In: *Algorithms and Computation* vol 1350 (1997), pp. 92–101.

Agenda

Frank-Wolfe Algorithm

Gradient descent method to find the global minimum of a convex function

Principle

At each iteration k, we dispose of a current continuous solution x_k :

1. Minimize linear approximation of Q around x_k according to its 1st order Taylor expansion
   - ▷ $b_k = \text{argmin}_{b \in \mathbb{R}^{n \times m}} \{b \nabla Q(x_k)\}$
2. Step size determination by a line search
   - ▷ $\gamma^* = \min_{\gamma \in [0;1]} \{Q(x_k + \gamma(b_k - x_k))\}$
3. Update current solution
   - ▷ $x_{k+1} = x_k + \gamma(b_k - x_k)$

Frank-Wolfe Algorithm for GED estimation

Relaxation of the binary constraint

Adaptation for GED

At each iteration k, we dispose of a current continuous solution x_k :

1. Minimize linear approximation of Q around x_k according to its 1st order Taylor expansion
   - ▷ $b_k = \text{argmin}_{b \in \mathcal{D}_{m,n}} \{b \nabla Q(x_k)\}$
   - $\Leftrightarrow$ $b_k = \text{argmin}_{b \in \Pi_{m,n}} \{b \nabla Q(x_k)\}$ $\rightarrow$ Becomes an LSAP
2. Step size determination by a line search
   - ▷ $\gamma^* = \min_{\gamma \in [0;1]} \{Q(x_k + \gamma(b_k - x_k))\}$
3. Update current solution
   - ▷ $x_{k+1} = x_k + \gamma(b_k - x_k)$

▷ After convergence, return to the discrete space by running **step 1**

Frank-Wolfe Algorithm

A simple procedure to find a local minimum of Q, if not convex

- ▷ Method used in context of Graph Matching (IPFP) [Leordeanu et al. 2009]
- ▷ Also used for GED estimation [Bougleux et al. 2017]
- ▷ Converge generally to a local minimum
- ▷ Strongly impacted by the initialization

IPFP : Impact of the initialization

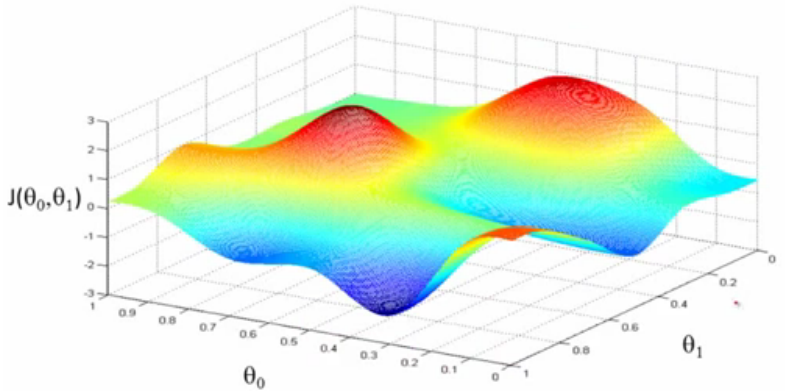


Figure: Example of a non-convex function

Non-convexity of Q

IPFP converges to a local minimum of Q. If Q is not convex, this minimum is not necessarily global.

- ▷ **Impact on time complexity** : if x_1 is far from a local minimum of Q, there may be more iterations to converge
- ▷ **Impact on accuracy** : if Q is non-convex, the returned local minimum depends on x_1

Proposal : mIPFP

A parallel multistart approach based on IPFP.

Procedure

1. Generate a set of k assignments S_k
2. Compute the set of refinements $\{IPFP(\mathbf{x}) \mid \mathbf{x} \in S_k\}$
3. Return $\text{mIPFP}(S_k) = \min_{\mathbf{x} \in S_k}\{Q(\mathbf{y}) : \mathbf{y} = \text{IPFP}(\mathbf{x})\}$

Proposal : mIPFP

A parallel multistart approach based on IPFP.

Procedure

1. Generate a set of k assignments S_k
2. Compute the set of refinements $\{IPFP(\mathbf{x}) \mid \mathbf{x} \in S_k\}$
3. Return $\text{mIPFP}(S_k) = \min_{\mathbf{x} \in S_k}\{Q(\mathbf{y}) : \mathbf{y} = \text{IPFP}(\mathbf{x})\}$

- ▷ Simple procedure
- ▷ Can be easily parallelized, as each IPFP is independent
- ▷ Several kinds of initializations are possible

mIPFP : Considered Initializations

Experiments are performed with the following initial matchings :

Optimal bipartite assignments

S_k is generated by the mbGED procedure

Low-cost bipartite assignments

S_k is generated by a greedy algorithm approximating the LSAP

Random assignments

S_k contains k assignments generated randomly without repetition, following a uniform distribution.

mIPFP : Considered Initializations

Experiments are performed with the following initial matchings :

Optimal bipartite assignments

S_k is generated by the mbGED procedure

Low-cost bipartite assignments

S_k is generated by a greedy algorithm approximating the LSAP

Random assignments

S_k contains k assignments generated randomly without repetition, following a uniform distribution.

These generators maintain the condition $k < l \Rightarrow S_k \subseteq S_l$, hence :

$$\text{mIPFP}(S_k) \geq \text{mIPFP}(S_{k+1}) \geq \text{mIPFP}(S_{k_{max}}) \tag{7}$$

Agenda

Datasets

Table: Characteristics of the four GREYC's chemistry datasets.

Dataset	*Number of graphs*	*Avg Size*	*Avg Degree*
Alkane	150	8.9	1.8
Acyclic	183	8.2	1.8
MAO	68	18.4	2.1
PAH	94	20.7	2.4
MUTA	100	$\approx$30	$\approx$2

Results

GED

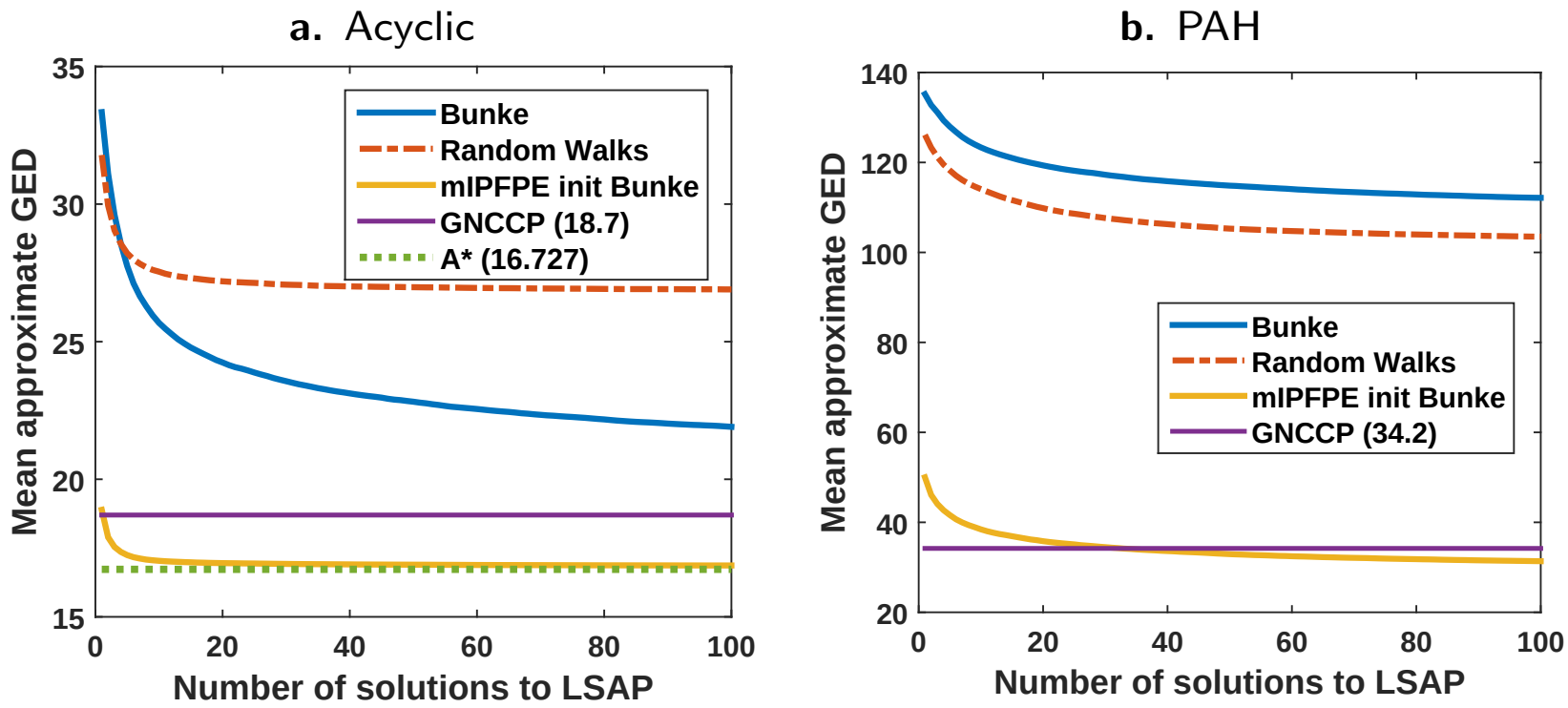


Figure: Approximate GED w.r.t the number of considered matchings

Experiments - Chemistry datasets

Bipartite and multiple bipartite GED estimations :

Algorithm	Alkane			Acyclic			PAH	
	d	e	t	d	e	t	d	t
A^*	15.3			16.7				
bGED-Stars	37.8	22.5	$\approx 10^{-4}$	33.3	16.6	$\approx 10^{-4}$	135.2	10^{-3}
bGED-RW	36.0	20.7	0.02	31.8	15.0	0.02	125.8	2.60
mbGED-Stars	**25.5**	**10.2**	0.01	**23.1**	**6.4**	0.01	116.0	0.02
mbGED-RW	26.0	10.6	0.03	27.0	10.3	0.02	**105.8**	2.65
mbGED Greedy	39.0	23.6	$\approx 10^{-4}$	38.1	20.7	$\approx 10^{-4}$	135.7	$< 10^{-3}$
GNCCP	16.6	1.2	0.58	18.7	1.3	0.32	34.2	14.44

Table: Mean approximate GED (d), error (e) and computation time (t) $k = 40$

Experiments - Chemistry datasets

Frank-Wolfe and multistart Frank-Wolfe GED estimation :

Algorithm	Acyclic			Alkane			PAH	
	d	e	t	d	e	t	d	t
A^*	15.3			16.7				
$\text{IPFP}_{\text{Init.-Stars}}$	18.1	2.7	0.02	18.9	2.2	0.009	50.0	0.09
$\text{IPFP}_{\text{Init.-RW}}$	18.0	2.7	0.04	18.7	2.1	0.02	46.8	2.68
$\text{IPFP}_{\text{Random init.}}$	19.9	4.6	0.02	20.4	3.6	0.01	53.2	0.10
$\text{IPFP}_{\text{Greedy init.}}$	18.1	2.8	0.02	19.2	2.4	0.01	50.3	0.03
$\text{mIPFP}_{\text{Init Stars}}$	15.4	0.07	0.20	16.9	0.2	0.10	33.6	1.10
$\text{mIPFP}_{\text{Init RW}}$	15.6	0.2	0.18	17.4	0.7	0.08	**30.5**	3.58
$\text{mIPFP}_{\text{Init Rand}}$	**15.3**	$<$**0.01**	0.22	**16.74**	$<$**0.01**	0.13	36.6	1.17
$\text{mIPFP}_{\text{Init Greed}}$	15.4	0.06	0.17	16.8	0.03	0.11	35.4	1.01
GNCCP	16.6	1.2	0.58	18.7	1.3	0.32	34.2	14.44

Table: Mean approximate GED (d), error (e) and computation time (t) $k = 40$

Experiments

Bipartite and Frank-Wolfe GED estimations :

Algorithm	MAO		MUTA		CMU $k = 16$	
	d	t	d	t	d	t
bGED-Stars	95.7	0.001	209	0.02	1810	0.01
mbGED-Stars	75.1	0.05	**196**	0.9	**1791**	0.10
mbGED Greed	99.6	< 0.001	210	0.005	4418	0.02
$\text{IPFP}_{\text{Init. Stars}}$	38.4	0.04	136	0.07	410	0.07
$\text{IPFP}_{\text{Init Rand}}$	56.3	0.07	142	0.1	6532	0.90
$\text{IPFP}_{\text{Init Greedy}}$	38.8	0.02	136	0.09	414	0.02
$\text{mIPFP}_{\text{Init Stars}}$	**31.4**	0.46	**127**	3.25	410	1.35
$\text{mIPFP}_{\text{Init Rand}}$	33.3	0.81	128	5.20	472	13.64
$\text{mIPFP}_{\text{Init Greed}}$	31.8	0.46	133	3.58	415	0.7
GNCCP	34.3	9.23		-	**408**	6.66

Table: Mean approximate graph edit distance (d), error (e) and computation time (t), $k = 40$.

Agenda

Two conflicting criteria for a better generator

Quality of IPFP relies mostly on the quality of the initial solution.
How can we produce better initial solutions for a parallelized algorithm ?

Two conflicting criteria for a better generator

Quality of IPFP relies mostly on the quality of the initial solution.
How can we produce better initial solutions for a parallelized algorithm ?

A good solution generator should follow the two conflicting objectives:

- ▷ Producing solutions that are "far" from one another : **exploration criterion**
- ▷ Producing solutions that are already good solution in terms of GED : **quality criterion**

Our proposition for a better generator: RANDPOST(k,l)

The Algorithm we propose is a refinement of mIPFP : it consists in several iterations of mIPFP where each new iteration generates k new solutions in a stochastic fashion such that each assignment $(i \rightarrow j)$ is picked with a probability roughly equal to:

$$\Psi_{ij} = \frac{\#\text{refined solutions that include}(i \rightarrow j)}{\#\text{refined solutions}}$$

Our proposition for a better generator: RANDPOST(k,l)

The Algorithm we propose is a refinement of mIPFP : it consists in several iterations of mIPFP where each new iteration generates k new solutions in a stochastic fashion such that each assignment $(i \rightarrow j)$ is picked with a probability roughly equal to:

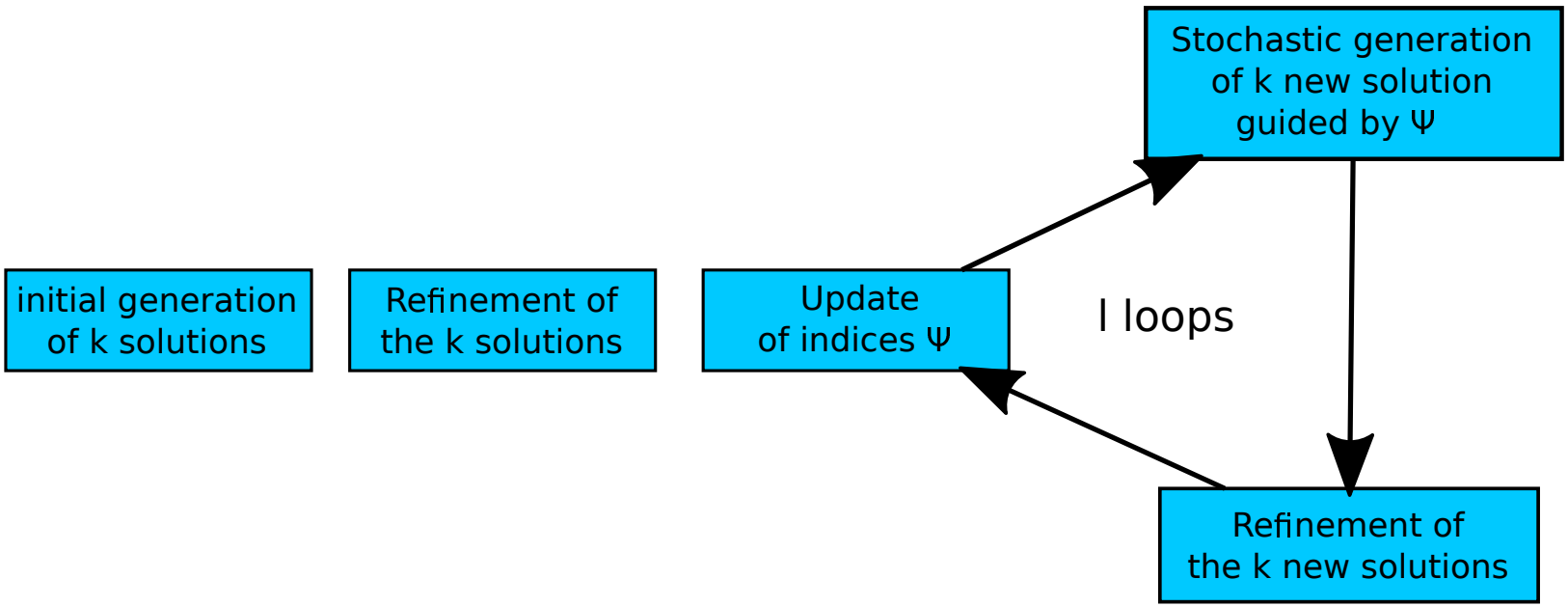
$$\Psi_{ij} = \frac{\#\text{refined solutions that include}(i \rightarrow j)}{\#\text{refined solutions}}$$

The randomness of the procedure answers the **exploration criterion**.

Pairwise assignments that are thought to appear in many good solutions are made more likely to be picked by the algorithm in order to answer the **quality criterion**

Our proposition for a better generator: RANDPOST(k,l)

General architecture of algorithm RANDPOST(k,l)



Agenda

Datasets

Dataset	*Number of graphs*	*Avg Size*
MAO	68	18.4
PAH	94	20.7
MUTA 10-70	10	10-70
ClinTox	25	115.7

▷ Monoamine Oxydase (MAO) dataset .This dataset is composed of 68 molecules divided into two classes: 38 molecules inhibit the monoamine oxidase (antidepressant drugs) and 30 do not.
▷ Polyciclic Aromatic Hydrocarbons dataset (PAH) This dataset is composed cyclic unlabeled graphs. All atoms are carbons, all bounds are aromatics. This is a classification problem (cancerous or not cancerous molecules).
▷ Mutagenicity Graphs (MUTA). Mutagen and non-mutagen molecules.
▷ ClinTox Dataset. Drugs approved by the FDA and those that have failed clinical trials for toxicity reasons.

Benchmark

Save for results on ClinTox dataset, our results were compared to the results of all 9 algorithms that participated to the Graph Distance Contest (ICPR 2016).

- absolute errors were computed w.r.t. to the best solutions found among all 13 algorithms (9 of contest + 4 versions of RANDPOST).

- For a given algorithm, "% best" represents the proportion of pairs of instance where the best GED among all 13 computed GEDs was found.

Experiments - metric costs

Algorithms	MAO				PAH				ClinTox			
	time	GED	err.	%best	time	GED	err.	%best	time	GED	err.	%best
RANDPOST(40, 1, 0)	0.013	34.43	10.30	25	0.013	36.94	24.82	1	3.542	209.42	52.12	0
RANDPOST(40, 40, 0)	0.074	24.16	0.03	98	0.099	21.23	9.11	19	17.205	167.76	10.46	2
RANDPOST(40, 20, 1)	0.029	**24.14**	**0.01**	**100**	0.038	20.71	8.59	27	13.330	163.18	5.88	10
RANDPOST(40, 10, 3)	0.051	24.19	0.06	98	0.063	**20.42**	**8.30**	**33**	19.514	160.24	2.94	30
RANDPOST(40, 5, 7)	0.144	24.48	0.35	89	0.116	20.90	8.78	26	29.278	**157.98**	**0.69**	**76**
Algorithms	MUTA 10				MUTA 20				MUTA 30			
	time	GED	err.	%best	time	GED	err.	%best	time	GED	err.	%best
RANDPOST(1, 0)	0.013	13.19	1.21	60	0.012	33.35	14.49	23	0.027	73.80	49.51	5
RANDPOST(40, 0)	0.020	**11.98**	**0.00**	**100**	0.080	19.00	0.14	86	0.235	25.68	1.39	42
RANDPOST(20, 1)	0.028	**11.98**	**0.00**	**100**	0.041	**18.96**	**0.10**	**91**	0.128	25.28	0.99	51
RANDPOST(10, 3)	0.062	**11.98**	**0.00**	**100**	0.062	19.03	0.17	89	0.181	**25.07**	**0.78**	**61**
RANDPOST(5, 7)	0.148	12.01	0.03	97	0.153	19.33	0.47	73	0.452	25.51	1.22	51
Algorithms	MUTA 40				MUTA 50				MUTA 60			
	time	GED	err.	%best	time	GED	err.	%best	time	GED	err.	%best
RANDPOST(1, 0)	0.063	83.94	50.23	2	0.123	81.67	44.83	5	0.246	98.55	51.97	5
RANDPOST(40, 0)	0.575	36.07	2.36	26	1.141	40.10	3.26	20	2.120	50.64	4.06	11
RANDPOST(20, 1)	0.302	35.00	1.29	46	0.565	38.56	1.72	31	1.158	48.95	2.37	24
RANDPOST(10, 3)	0.391	**34.31**	**0.60**	**67**	0.886	**37.57**	**0.73**	**61**	1.862	47.69	1.11	54
RANDPOST(5, 7)	0.516	34.85	1.14	53	1.465	37.84	1.00	55	3.133	**47.33**	**0.75**	**56**
Algorithms	MUTA 70				MUTA 100+				MUTAmix			
	time	GED	err.	%best	time	GED	err.	%best	time	GED	err.	%best
RANDPOST(1, 0)	0.528	84.18	25.80	6	3.181	259.28	37.88	0	0.111	155.71	21.68	6
RANDPOST(40, 0)	3.641	63.90	5.52	12	19.67	234.24	12.84	1	0.848	136.08	2.05	42
RANDPOST(20, 1)	2.559	61.45	3.07	25	12.39	227.49	6.09	9	0.455	135.16	1.13	57
RANDPOST(10, 3)	3.573	59.93	1.55	49	18.51	224.50	3.10	34	0.634	**134.75**	**0.72**	**68**
RANDPOST(5, 7)	8.181	**59.44**	**1.06**	**56**	28.70	**222.15**	**0.75**	**78**	1.117	135.06	1.03	55

Agenda

Conclusions and future work

Summary

- ▷ Allows to generate a great number of initial solutions in little time.
- ▷ Improvement w.r.t. simple multistart method, especially on graphs with 30+ nodes
- ▷ By design, less parallelizable than simple multistart.

Future work

1. Test the method with different kinds of initialization methods
2. Test different kinds of Ψ-based probability distributions
3. Make the algorithm choose which criterion (exploration or quality) to favor based on the Ψ indices.
4. Make the method more parallelizable