Extending Convolutional Neural Networks to Irregular Domains through Graph Inference

Bastien Pasdeloup

bastien.pasdeloup@imt-atlantique.fr

Ph.D. director: Dominique Pastor
Ph.D. advisor: Vincent Gripon

December 12, 2017

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

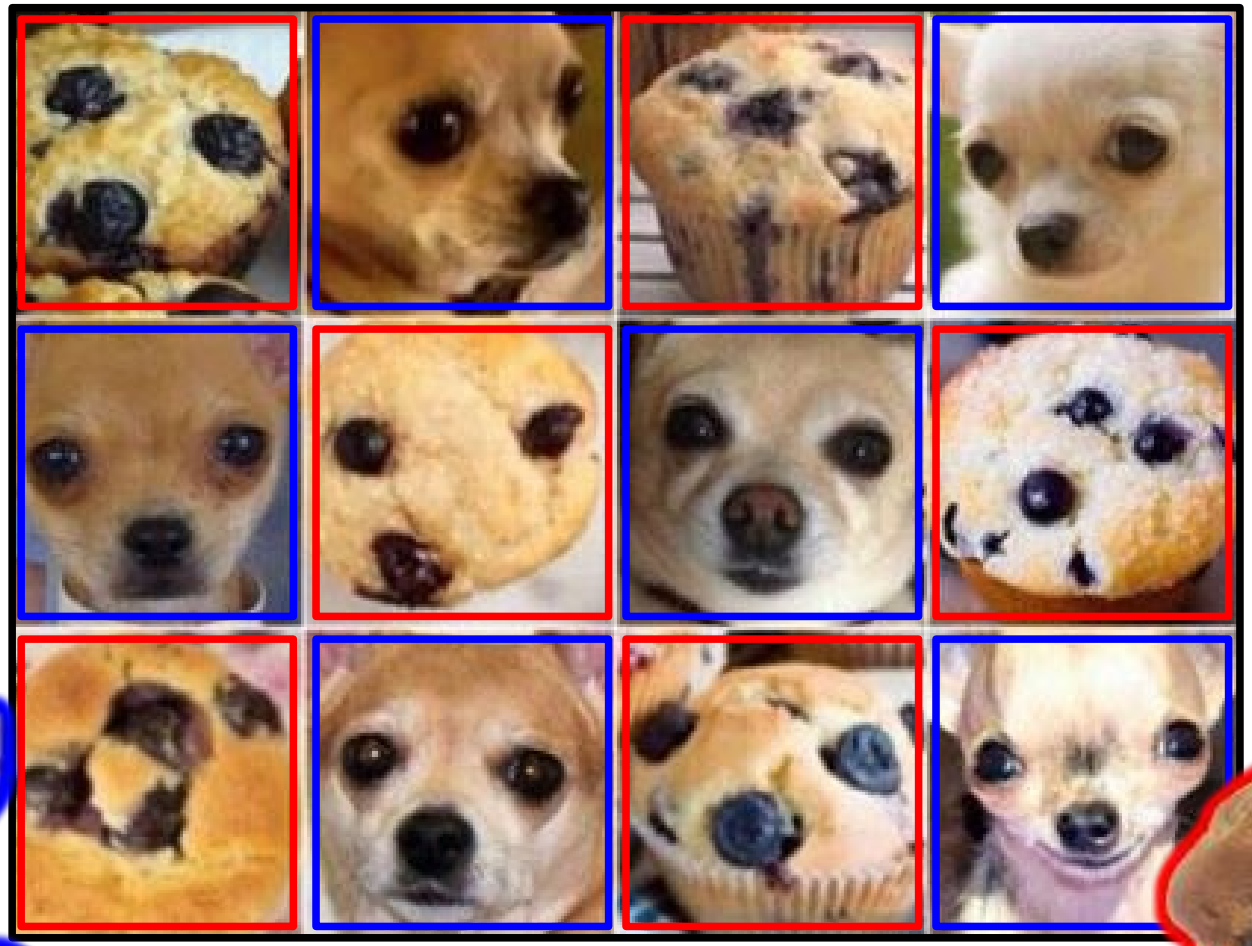
Lab-STICC

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

Introduction — The problem of signals classification



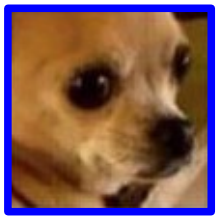
Introduction — The problem of signals classification







Introduction — Supervised classification (*eg.*: SVM)





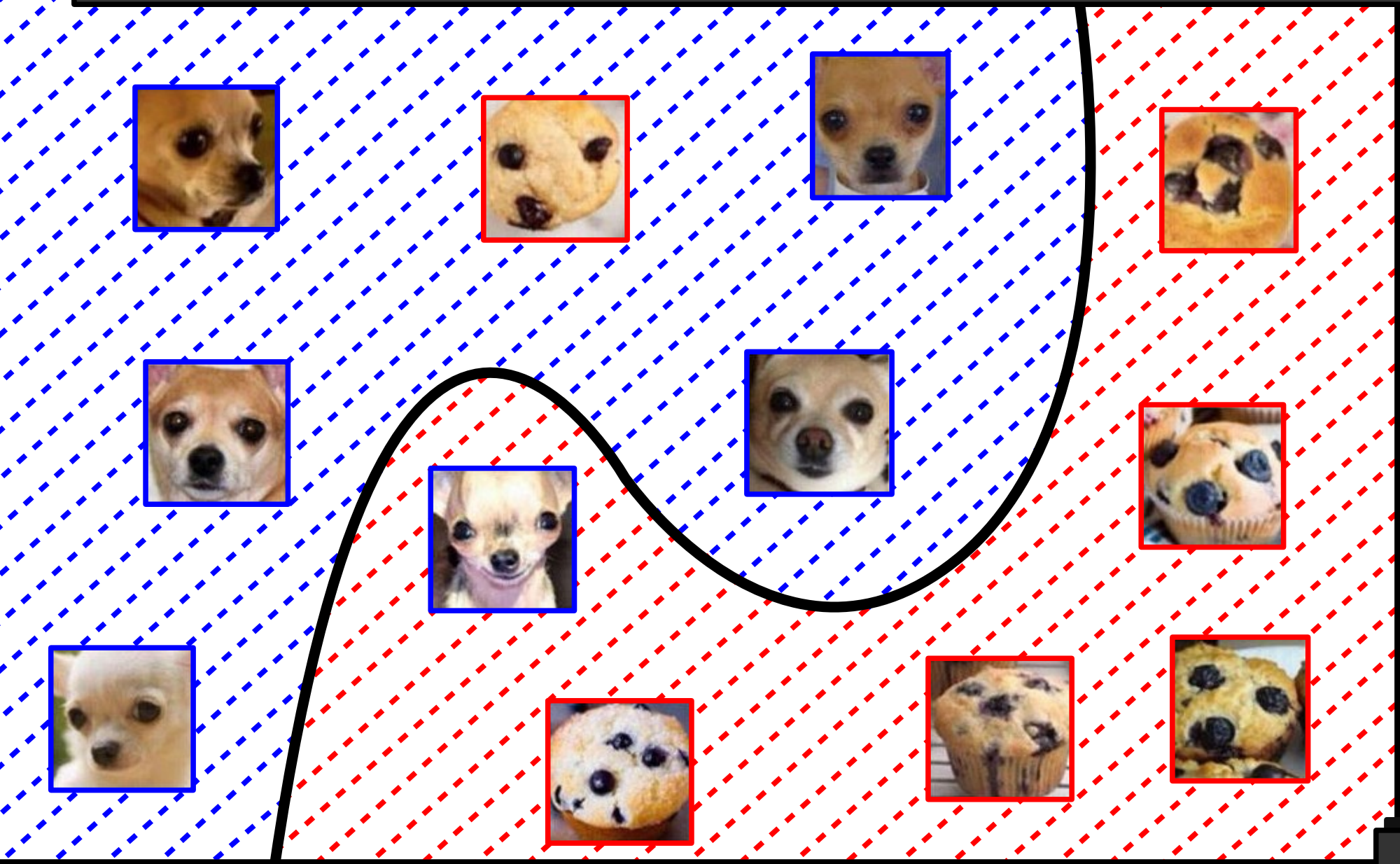




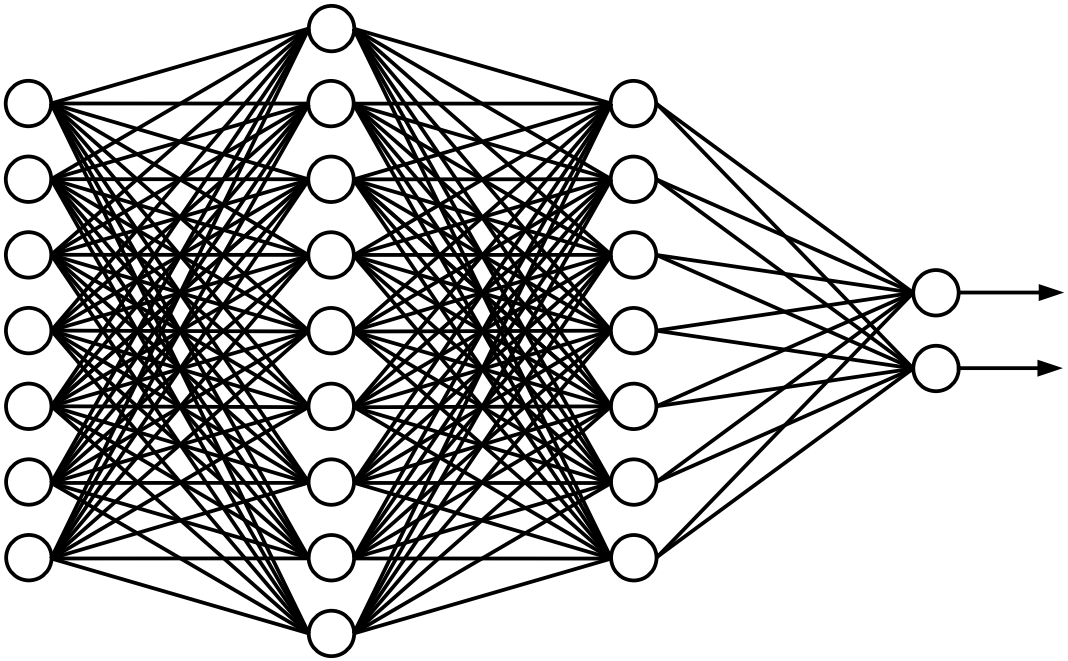




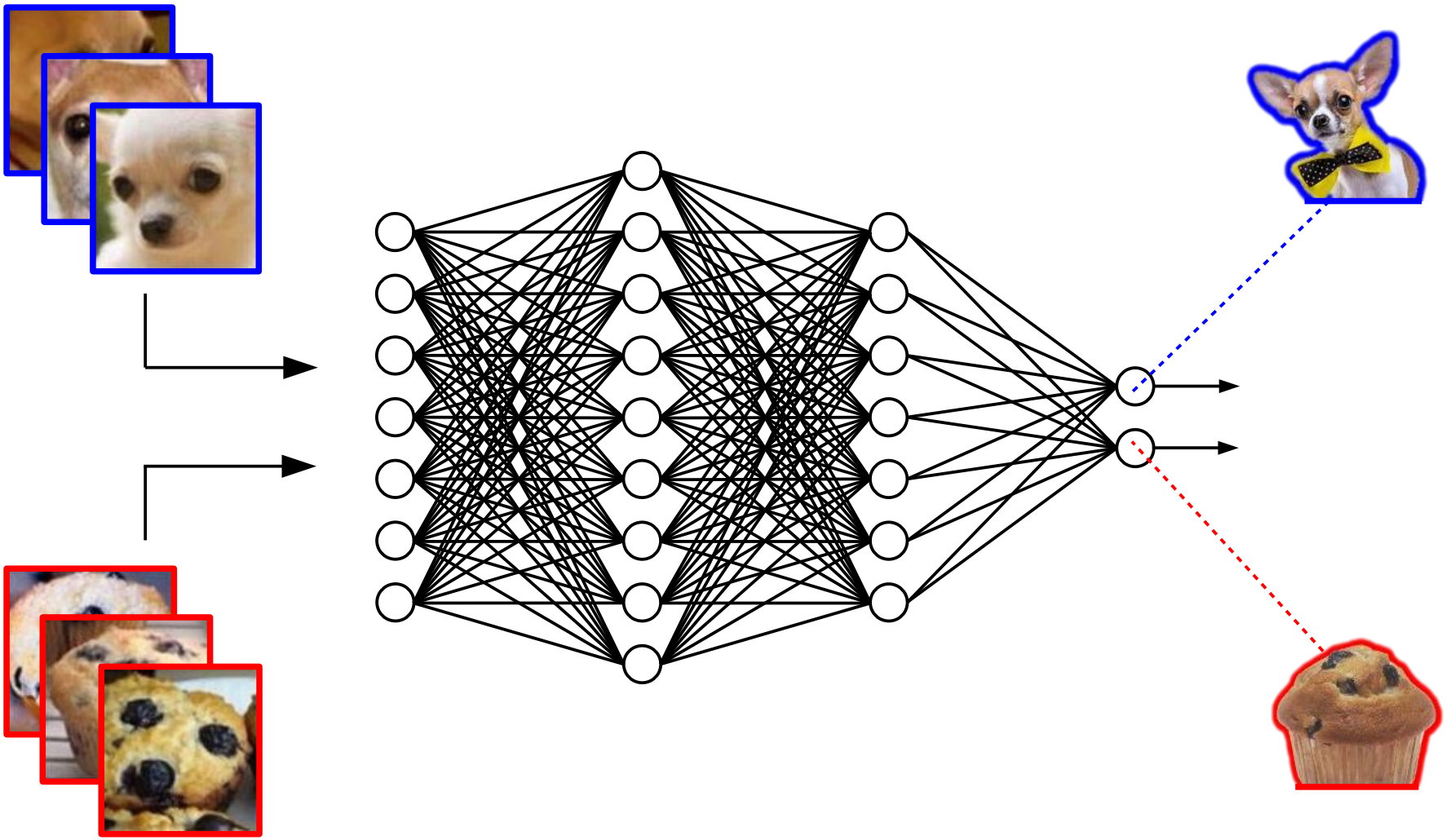

Introduction — Supervised classification (*eg.*: SVM)



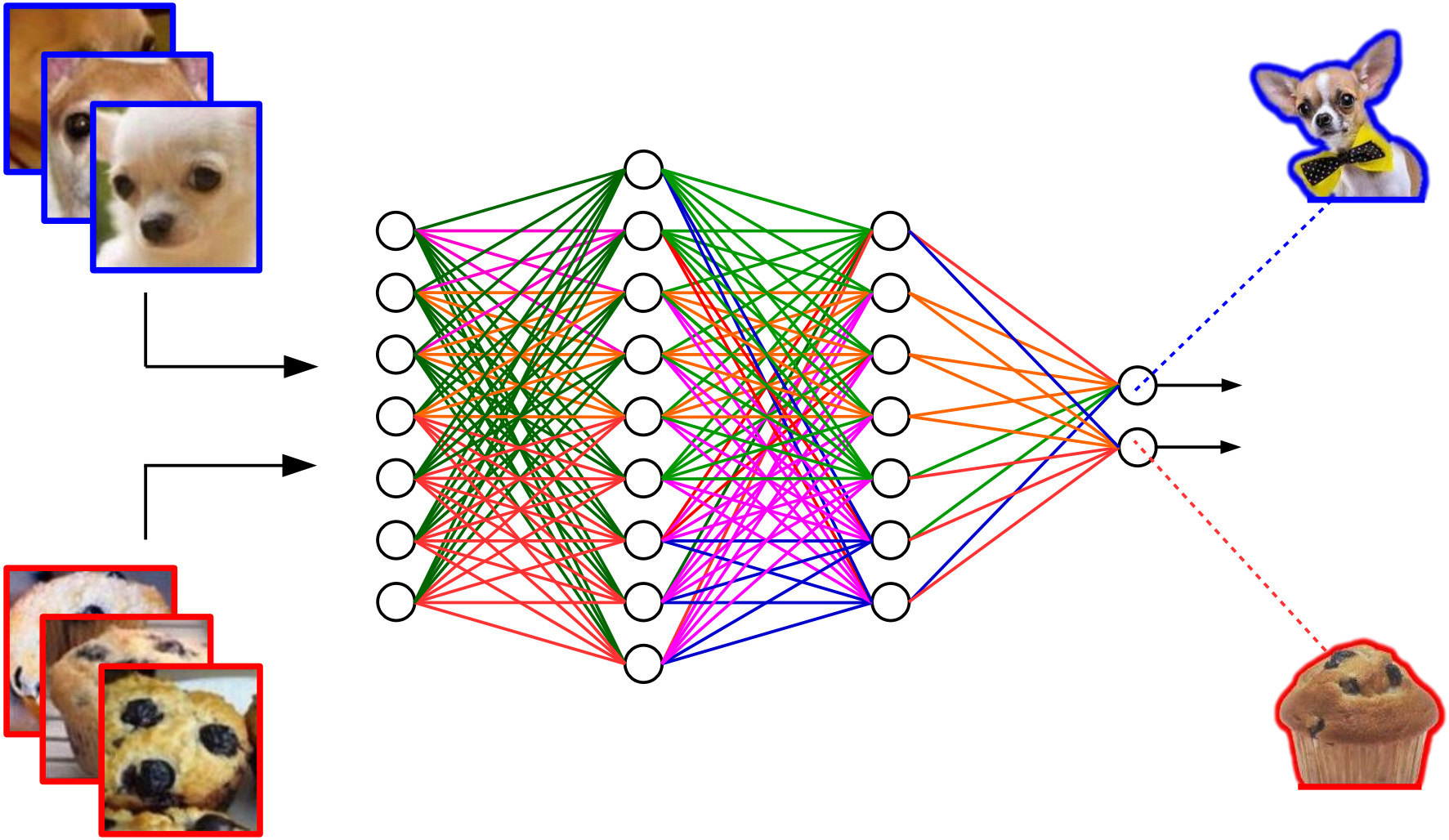
Introduction — Supervised classification (*eg.*: NN)



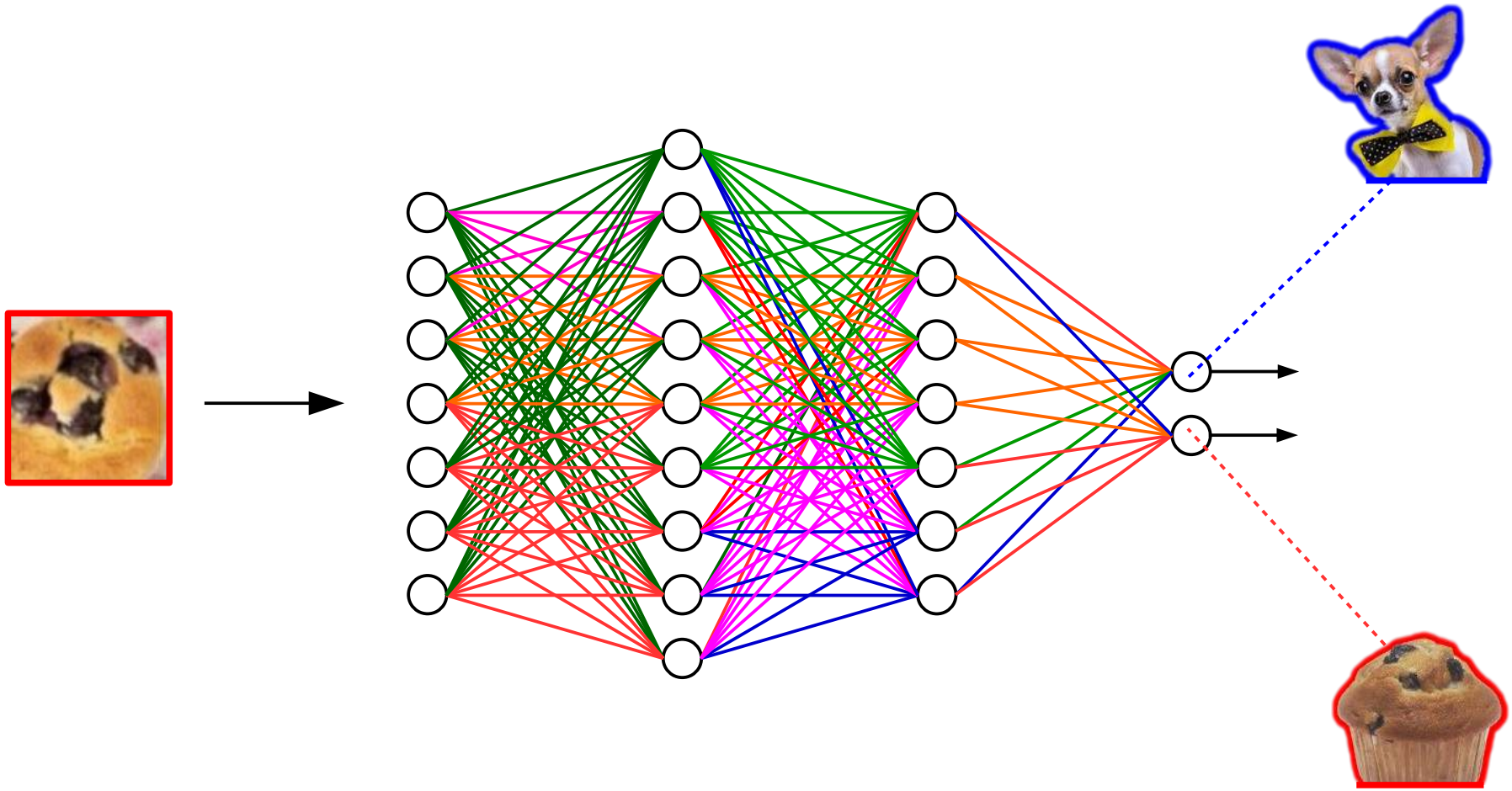
Introduction — Supervised classification (*eg.*: NN)



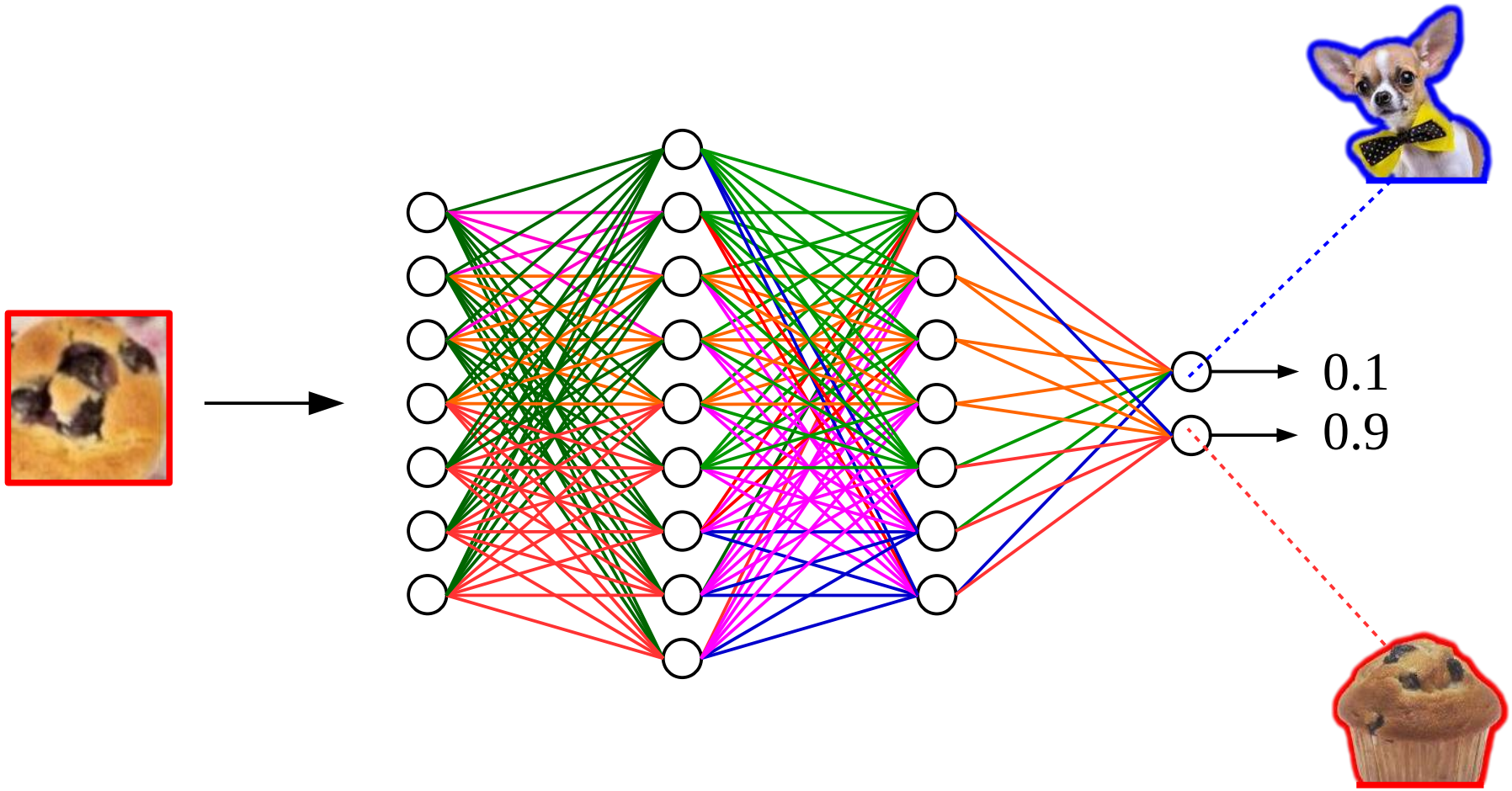
Introduction — Supervised classification (*eg.*: NN)



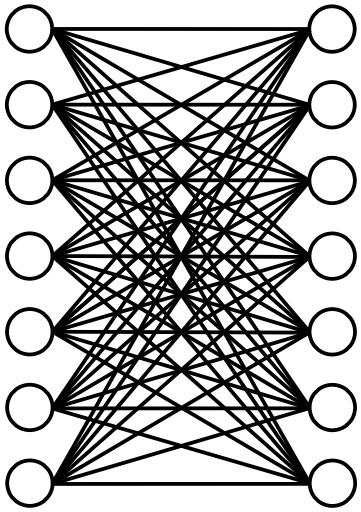
Introduction — Supervised classification (*eg.*: NN)



Introduction — Supervised classification (*eg.*: NN)

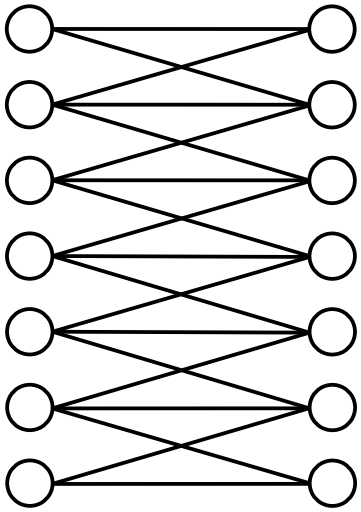


Introduction — From dense to convolutional layers



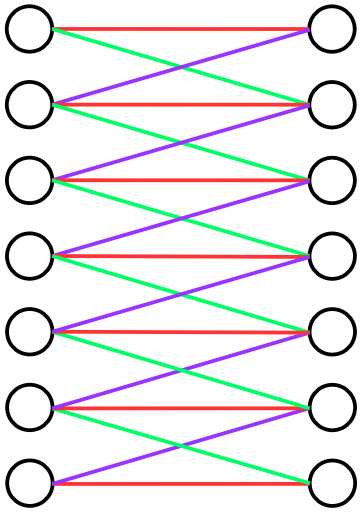
$$\begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} & w_{15} & w_{16} & w_{17} \\ w_{21} & w_{22} & w_{23} & w_{24} & w_{25} & w_{26} & w_{27} \\ w_{31} & w_{32} & w_{33} & w_{34} & w_{35} & w_{36} & w_{37} \\ w_{41} & w_{42} & w_{43} & w_{44} & w_{45} & w_{46} & w_{47} \\ w_{51} & w_{52} & w_{53} & w_{54} & w_{55} & w_{56} & w_{57} \\ w_{61} & w_{62} & w_{63} & w_{64} & w_{65} & w_{66} & w_{67} \\ w_{71} & w_{72} & w_{73} & w_{74} & w_{75} & w_{76} & w_{77} \end{bmatrix}$$

Introduction — From dense to convolutional layers



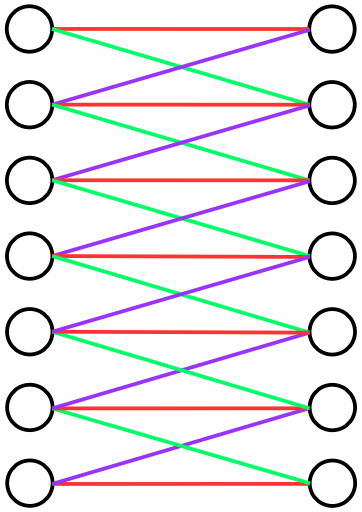
$$\begin{bmatrix} w_{11} & w_{12} & 0 & 0 & 0 & 0 & 0 \\ w_{21} & w_{22} & w_{23} & 0 & 0 & 0 & 0 \\ 0 & w_{32} & w_{33} & w_{34} & 0 & 0 & 0 \\ 0 & 0 & w_{43} & w_{44} & w_{45} & 0 & 0 \\ 0 & 0 & 0 & w_{54} & w_{55} & w_{56} & 0 \\ 0 & 0 & 0 & 0 & w_{65} & w_{66} & w_{67} \\ 0 & 0 & 0 & 0 & 0 & w_{76} & w_{77} \end{bmatrix}$$

Introduction — From dense to convolutional layers



$$\begin{bmatrix} w_{11} & w_{12} & 0 & 0 & 0 & 0 & 0 \\ w_{21} & w_{22} & w_{23} & 0 & 0 & 0 & 0 \\ 0 & w_{32} & w_{33} & w_{34} & 0 & 0 & 0 \\ 0 & 0 & w_{43} & w_{44} & w_{45} & 0 & 0 \\ 0 & 0 & 0 & w_{54} & w_{55} & w_{56} & 0 \\ 0 & 0 & 0 & 0 & w_{65} & w_{66} & w_{67} \\ 0 & 0 & 0 & 0 & 0 & w_{76} & w_{77} \end{bmatrix}$$

Introduction — From dense to convolutional layers



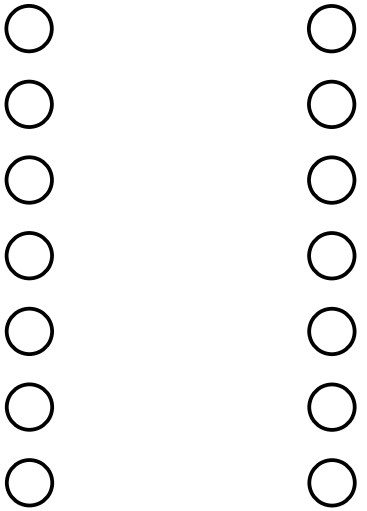
$$\begin{bmatrix} w_{11} & w_{12} & 0 & 0 & 0 & 0 & 0 \\ w_{21} & w_{22} & w_{23} & 0 & 0 & 0 & 0 \\ 0 & w_{32} & w_{33} & w_{34} & 0 & 0 & 0 \\ 0 & 0 & w_{43} & w_{44} & w_{45} & 0 & 0 \\ 0 & 0 & 0 & w_{54} & w_{55} & w_{56} & 0 \\ 0 & 0 & 0 & 0 & w_{65} & w_{66} & w_{67} \\ 0 & 0 & 0 & 0 & 0 & w_{76} & w_{77} \end{bmatrix}$$



Locality: *objects* lie on adjacent vertices
Weight-sharing: *objects* are similar wherever they are

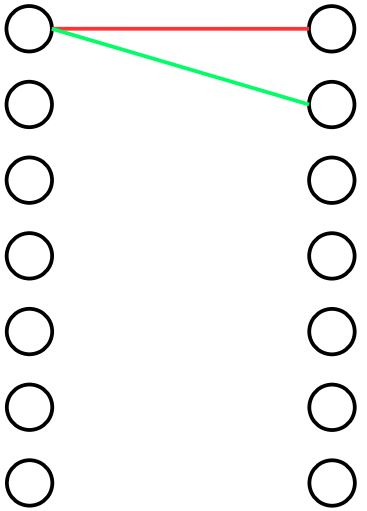
Introduction — The idea behind convolutional layers





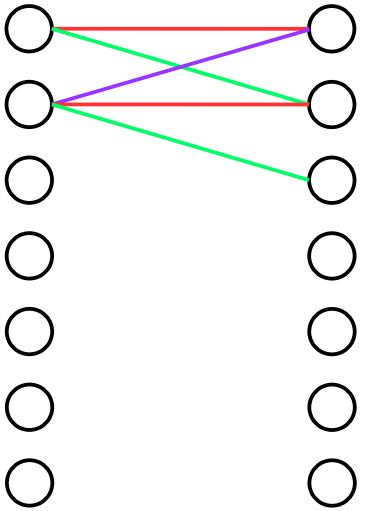
Introduction — The idea behind convolutional layers





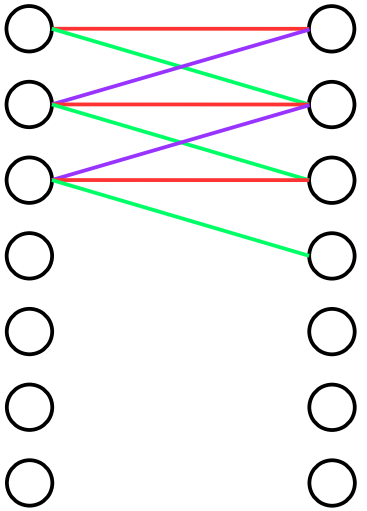
Introduction — The idea behind convolutional layers





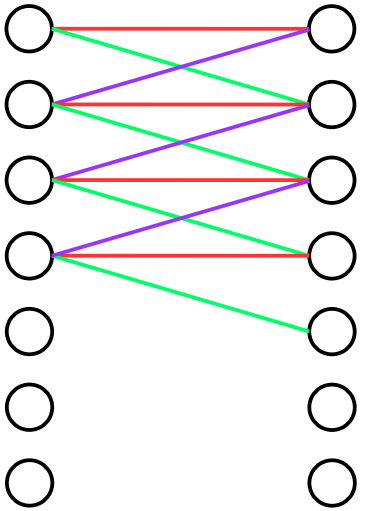
Introduction — The idea behind convolutional layers





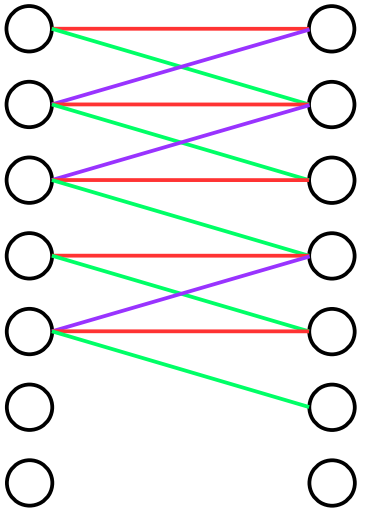
Introduction — The idea behind convolutional layers



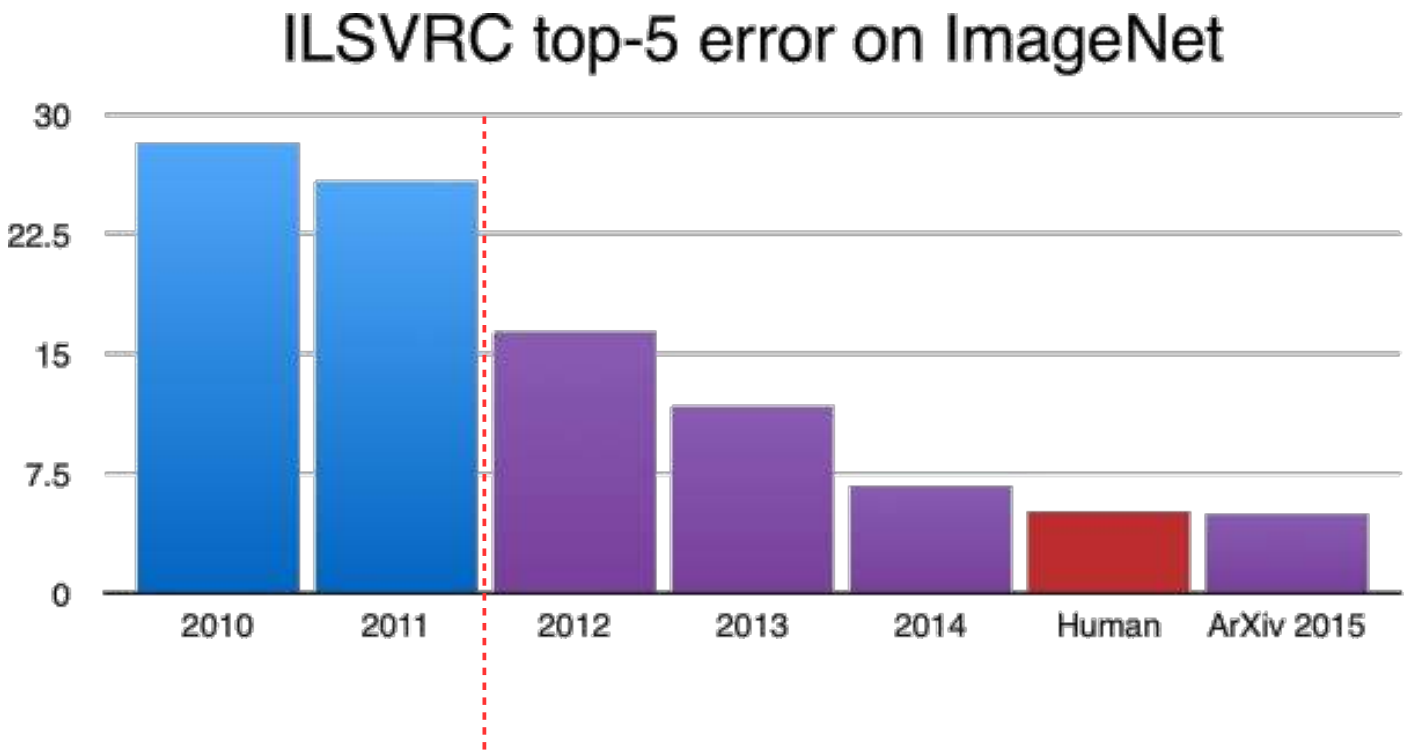


Introduction — The idea behind convolutional layers



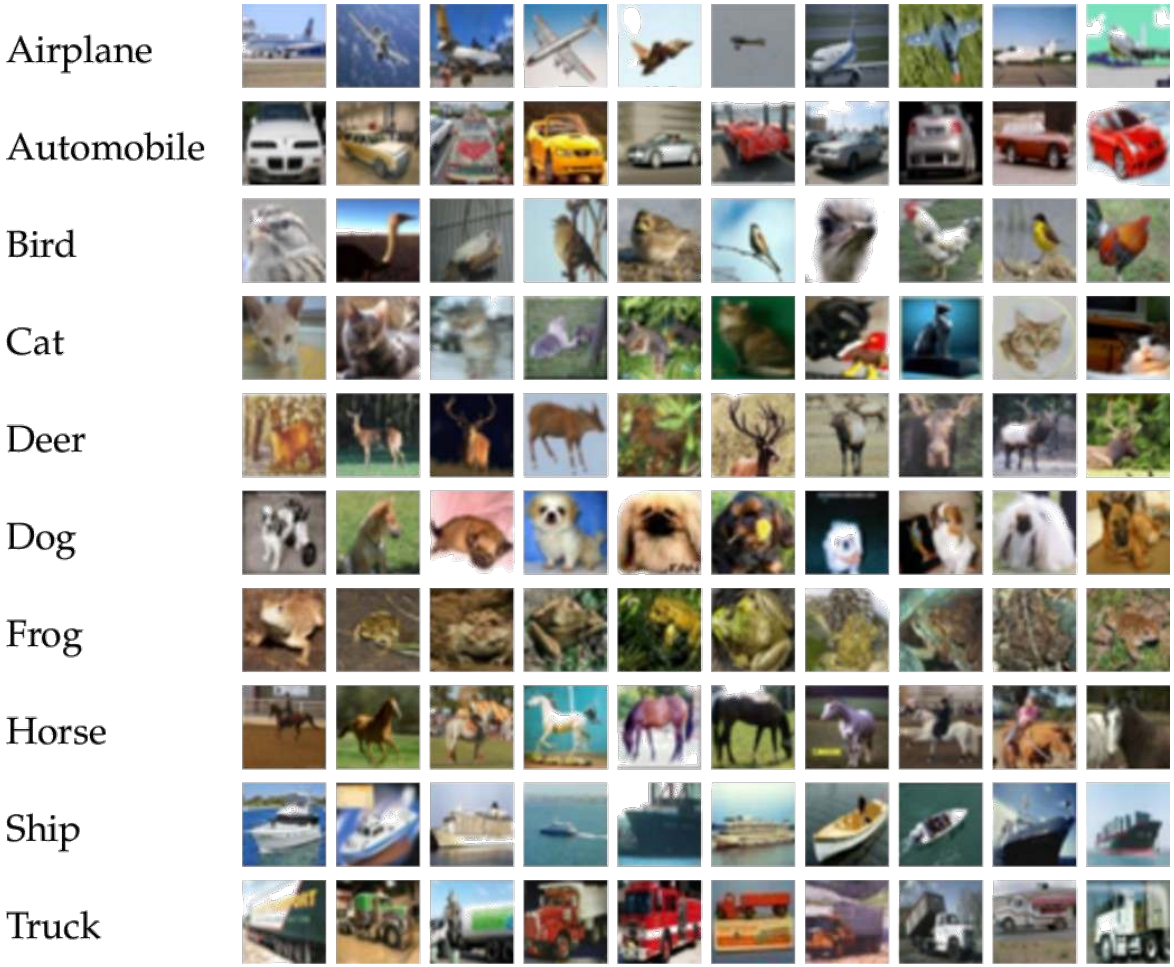


Introduction — Outstanding performance of deep neural networks



https://devblogs.nvidia.com/parallelforall/mocha-jl-deep-learning-julia/

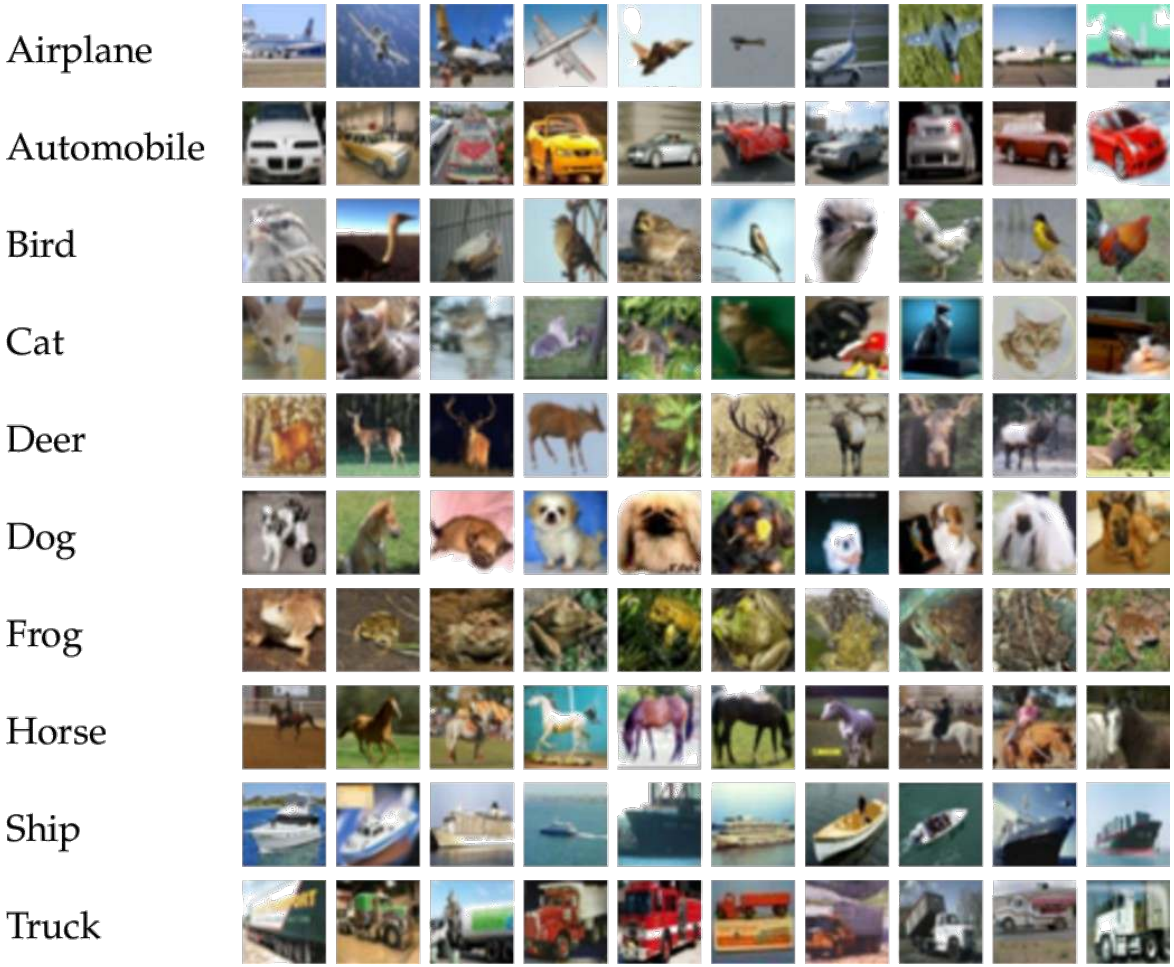
Introduction — Outstanding performance of CNNs



Dense
72.7 %

Cheng *et al.* — Deep Haar scattering networks (2016)

Introduction — Outstanding performance of CNNs

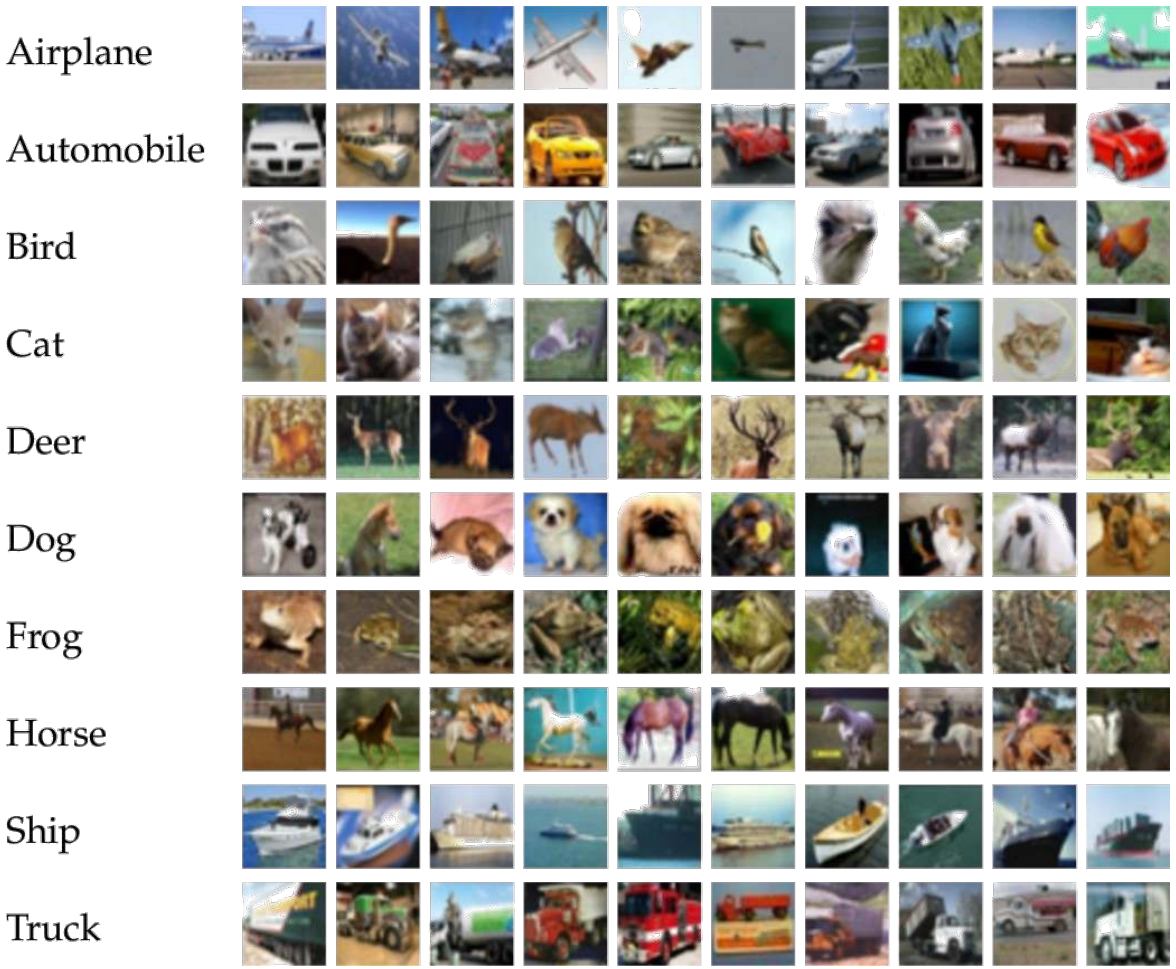


Dense
72.7 %

Convolutional
94.81 %

Huang *et al.* — Densely connected convolutional networks (2016)

Introduction — Outstanding performance of CNNs



Dense
72.7 %

Convolutional
94.81 %

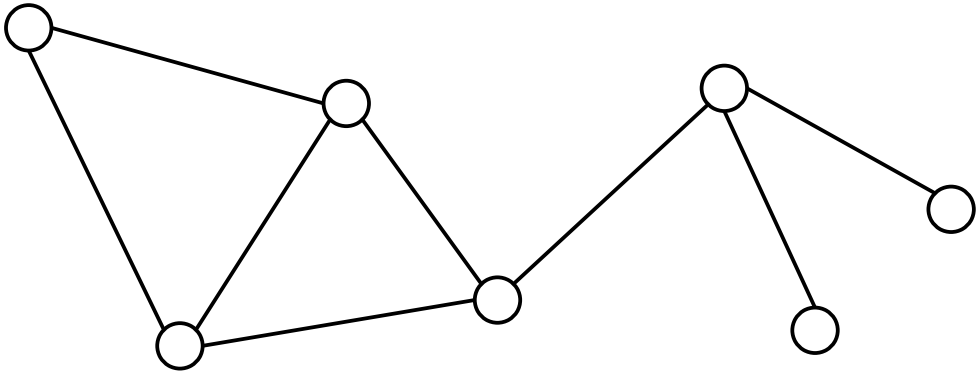
Data augmentation
97.28 %

Dutt *et al.* — Coupled ensembles of neural networks (2017)

Introduction — What about irregular data?



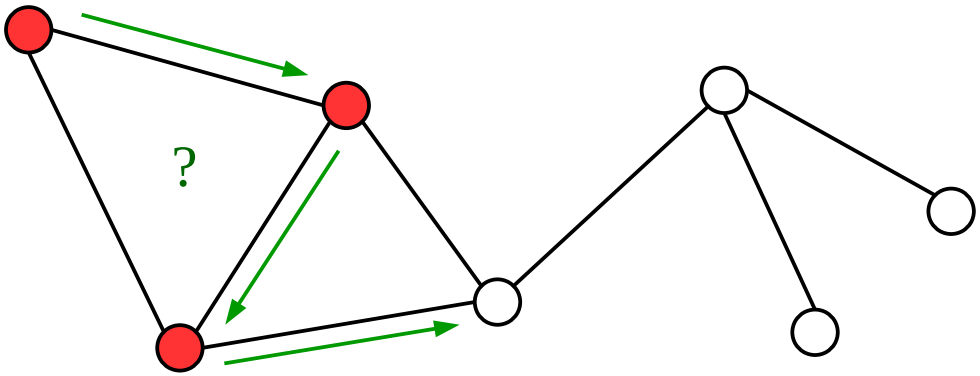




Introduction — What about irregular data?







Outline

Existing methods & required notions

The method we propose

— Global idea of the method —
— Graph inference from data —
— Identification of translations —

Results

Outline

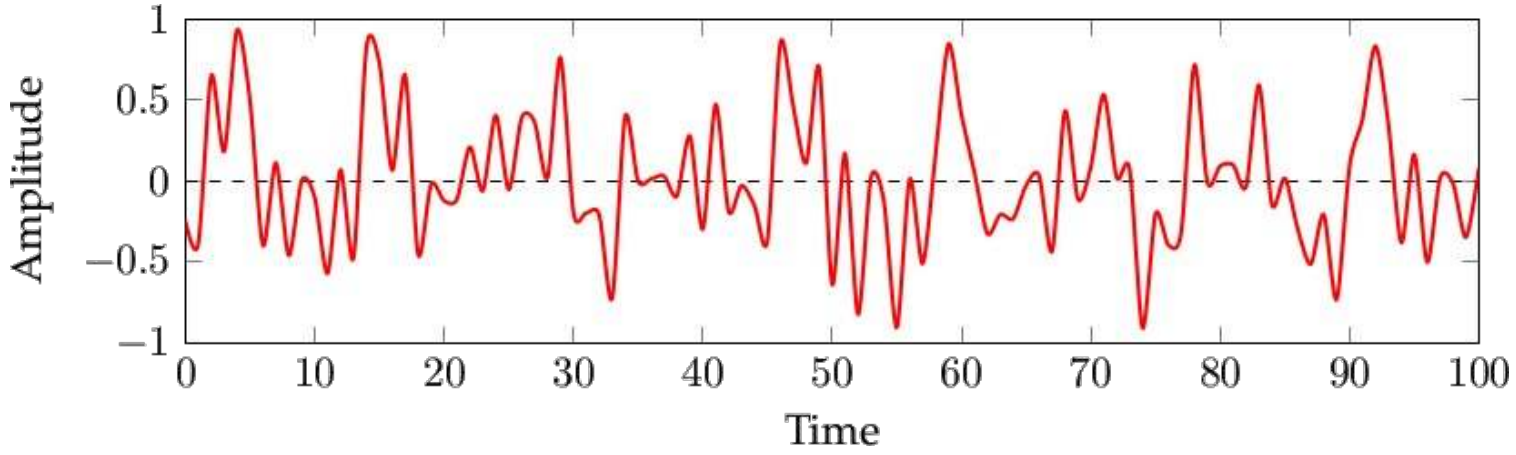
Existing methods & required notions

The method we propose

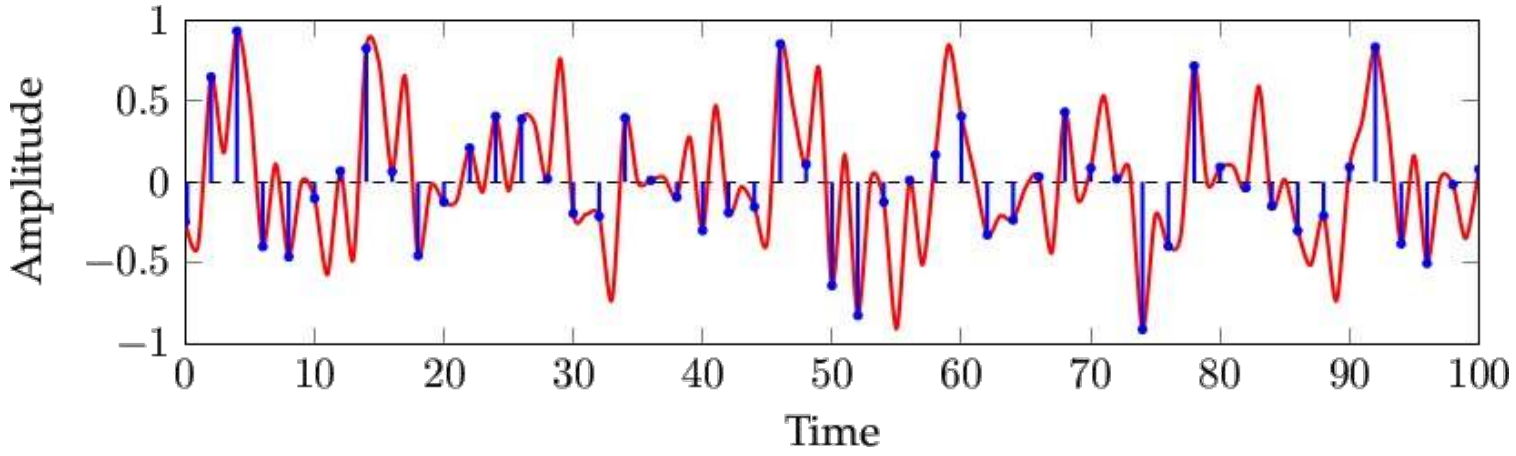
— Global idea of the method —
— Graph inference from data —
— Identification of translations —

Results

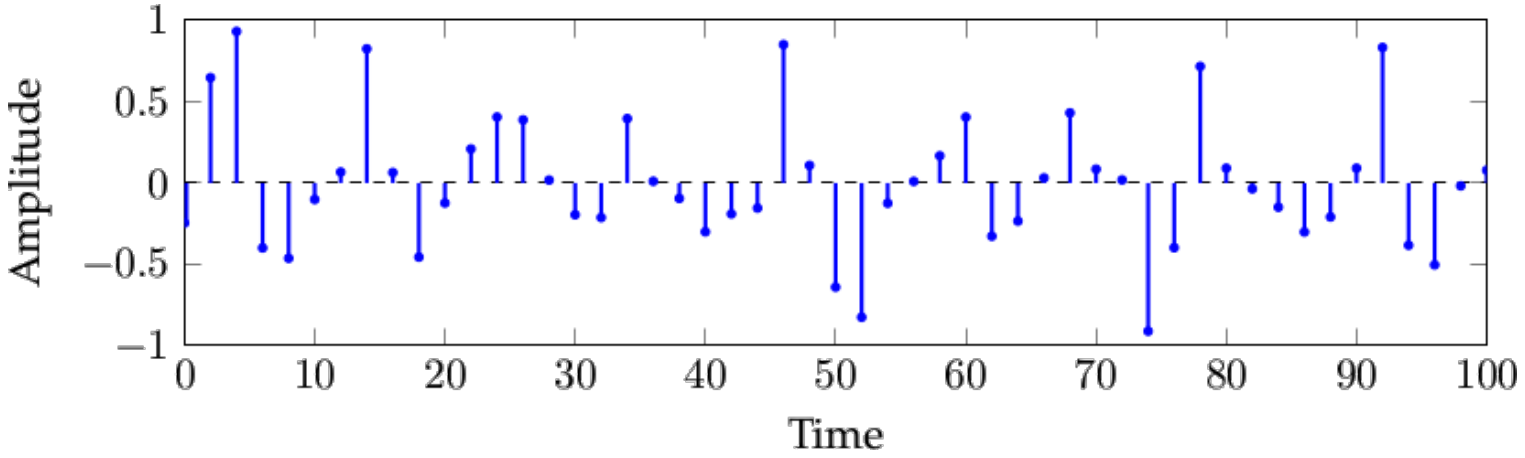
Methods & notions — The origins of GSP



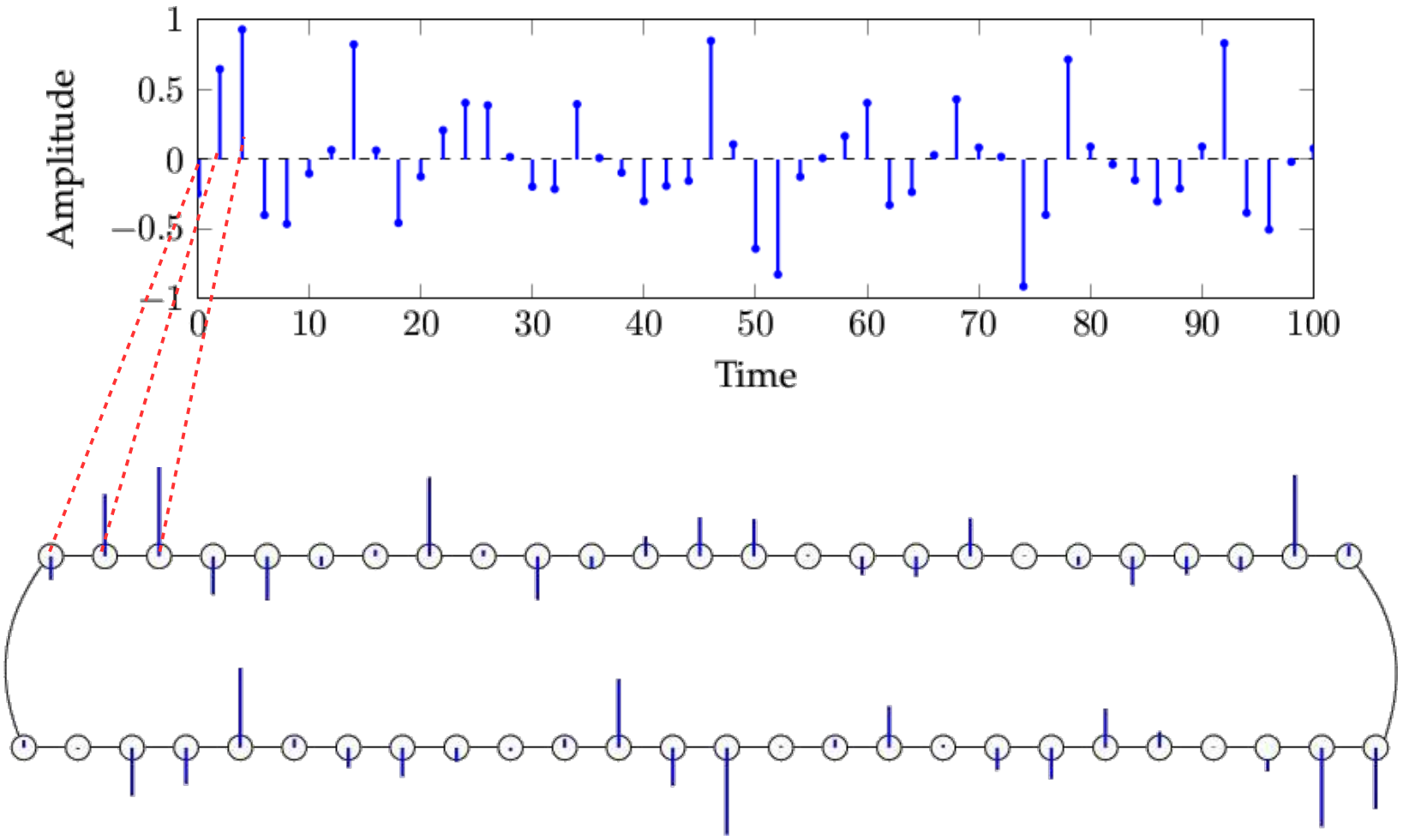
Methods & notions — The origins of GSP



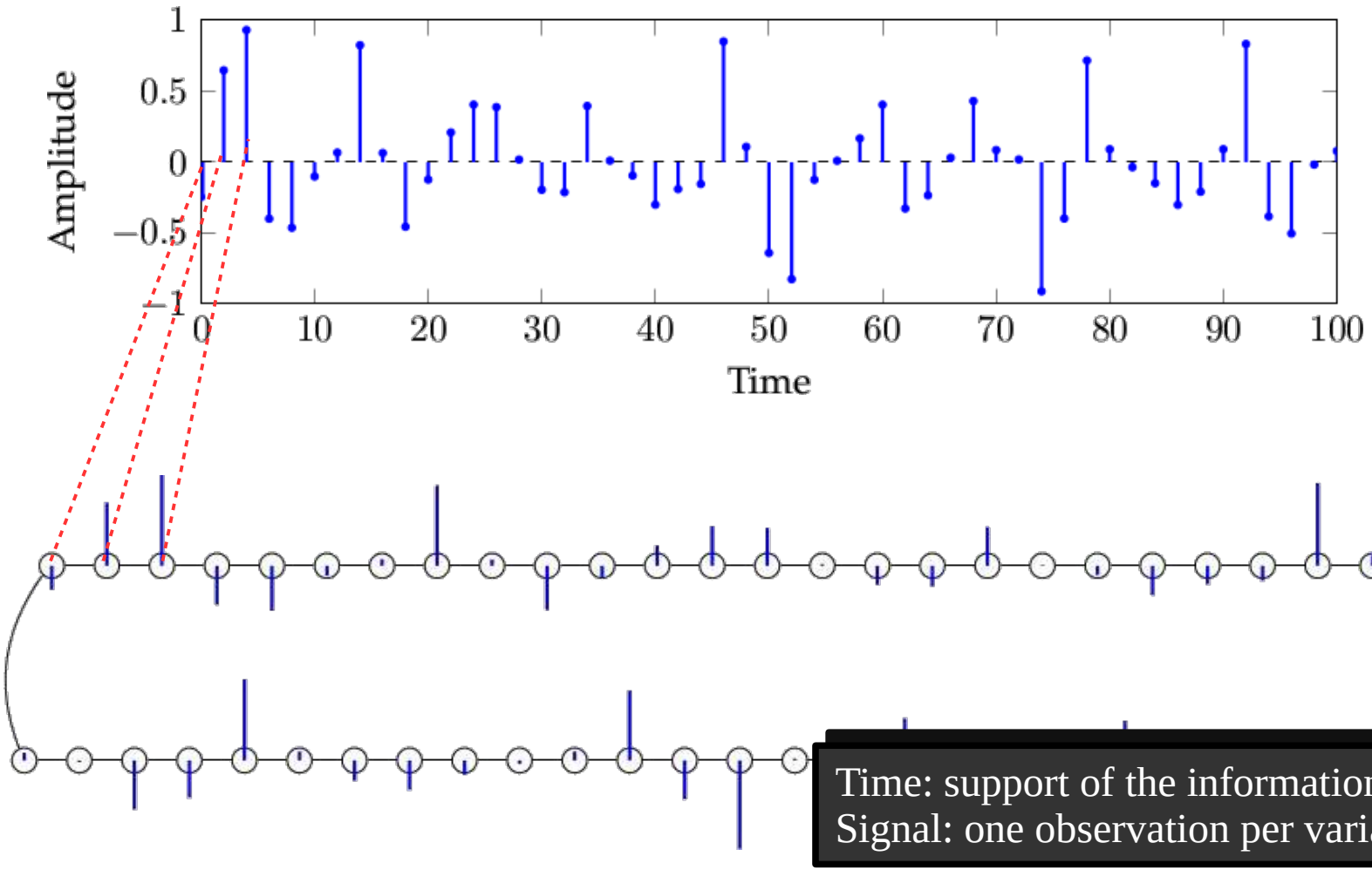
Methods & notions — The origins of GSP



Methods & notions — The origins of GSP

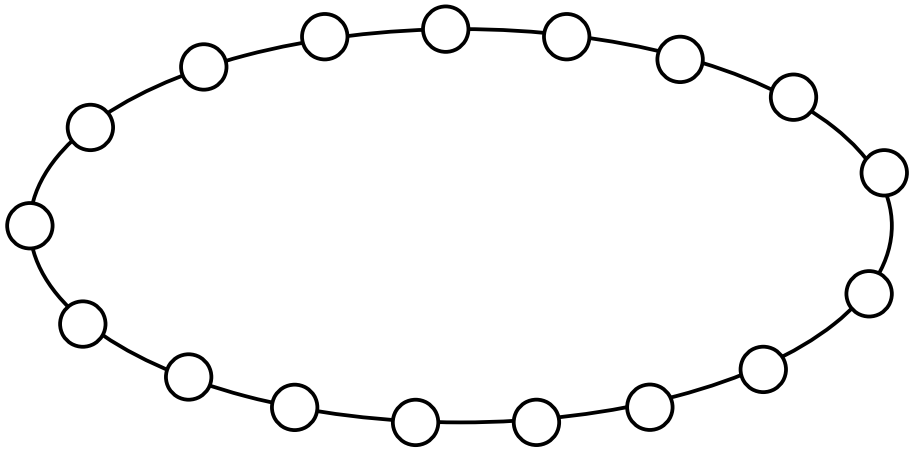


Methods & notions — The origins of GSP

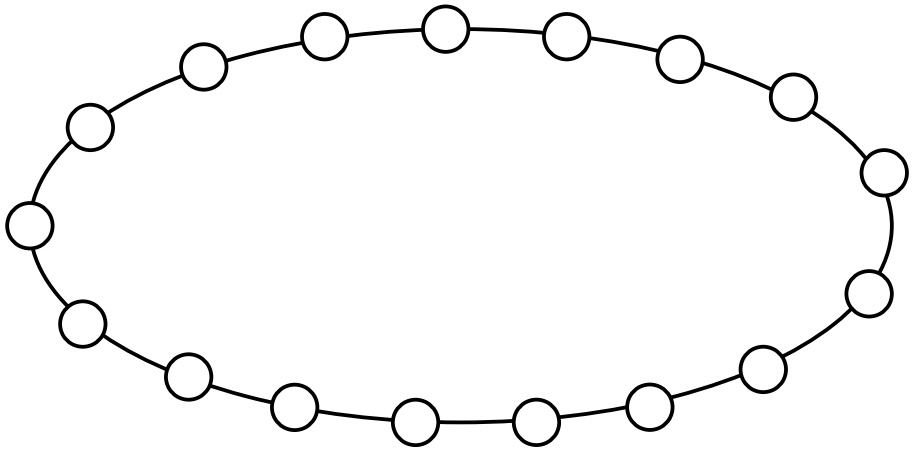


Time: support of the information
Signal: one observation per variable

Methods & notions — The origins of GSP

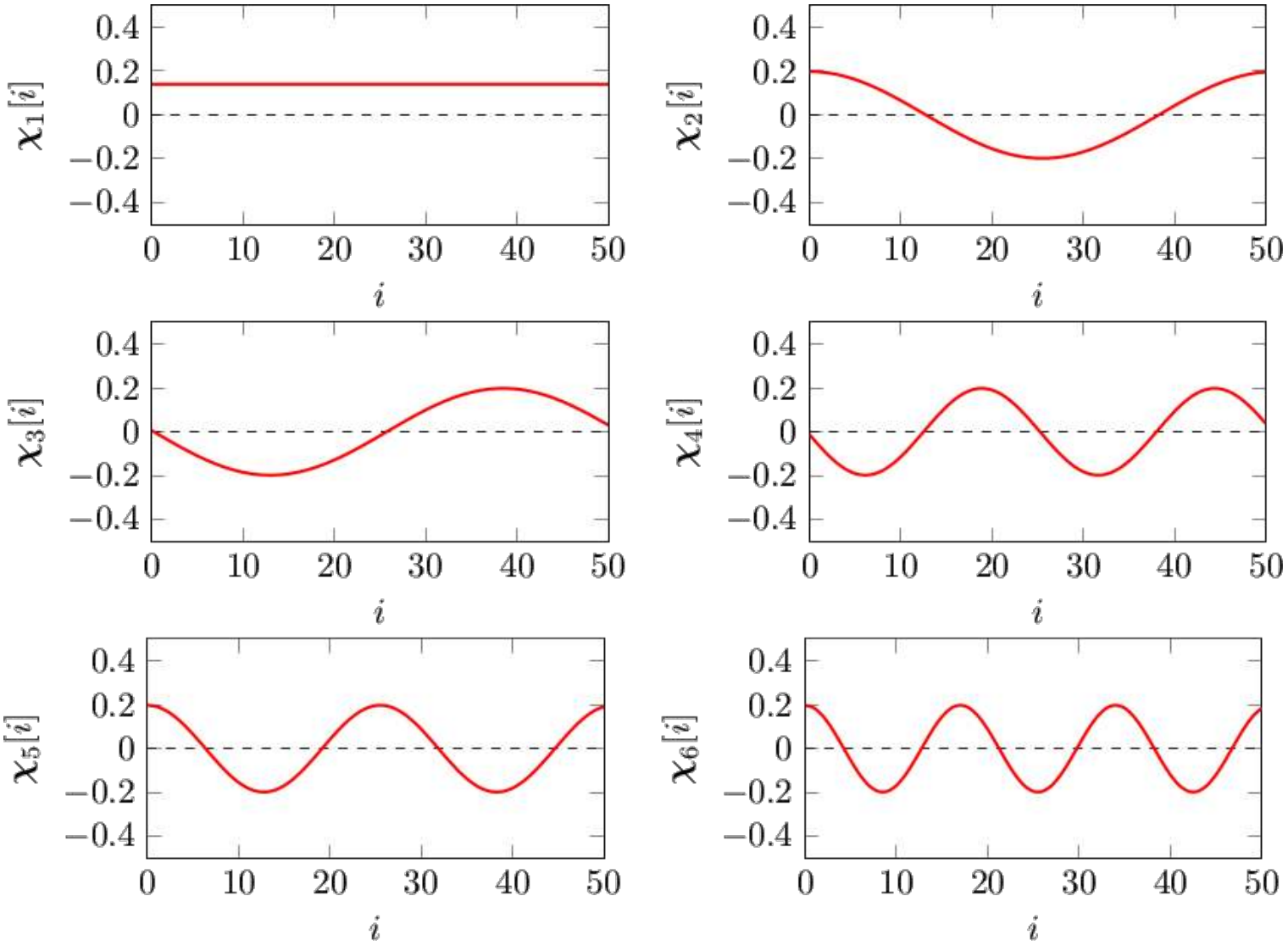


Methods & notions — The origins of GSP

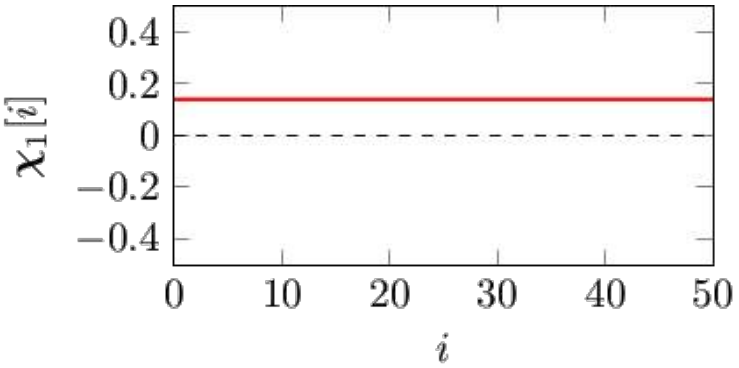


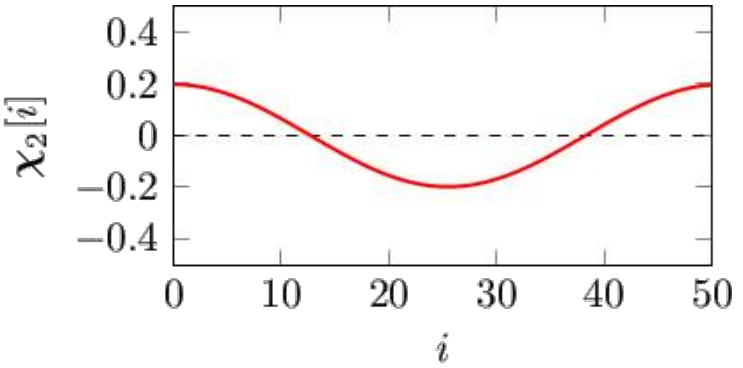
$$\begin{bmatrix}
2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
-1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2
\end{bmatrix}$$

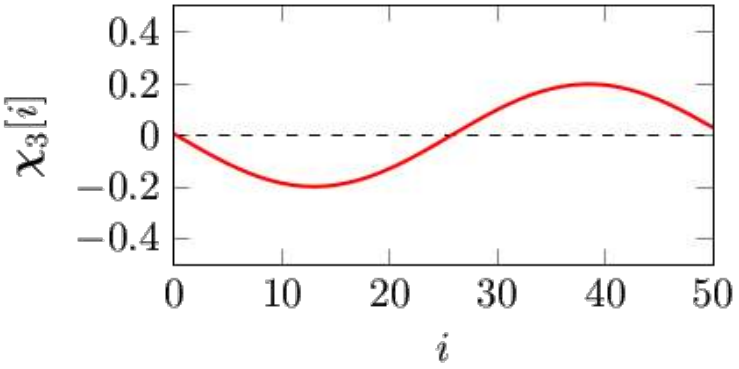
Methods & notions — The origins of GSP

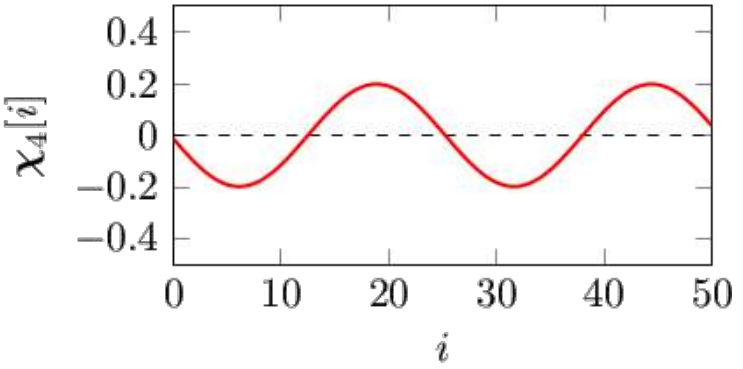


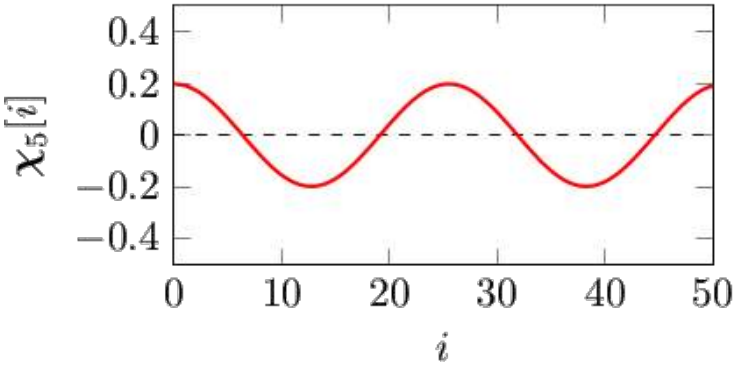
Methods & notions — The origins of GSP

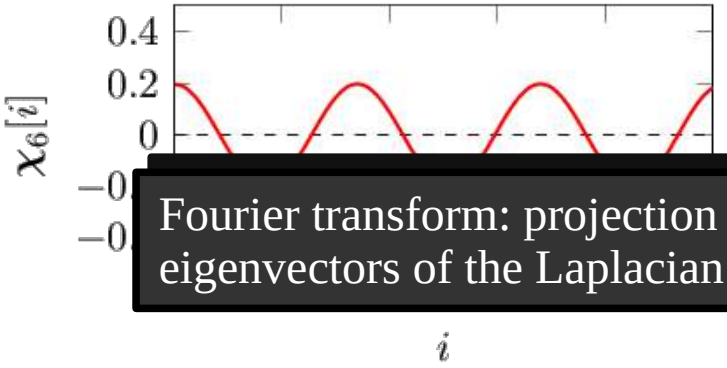






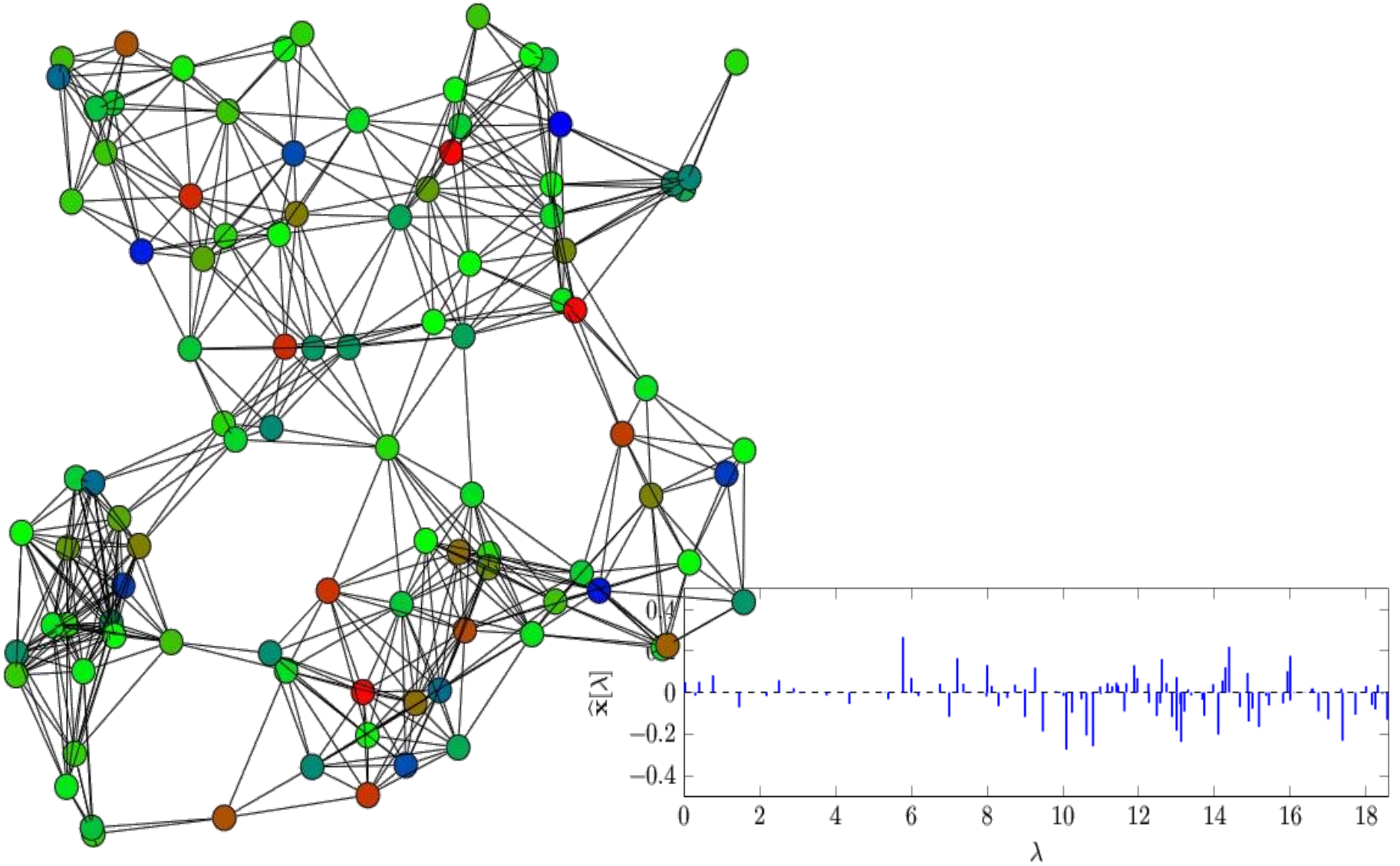




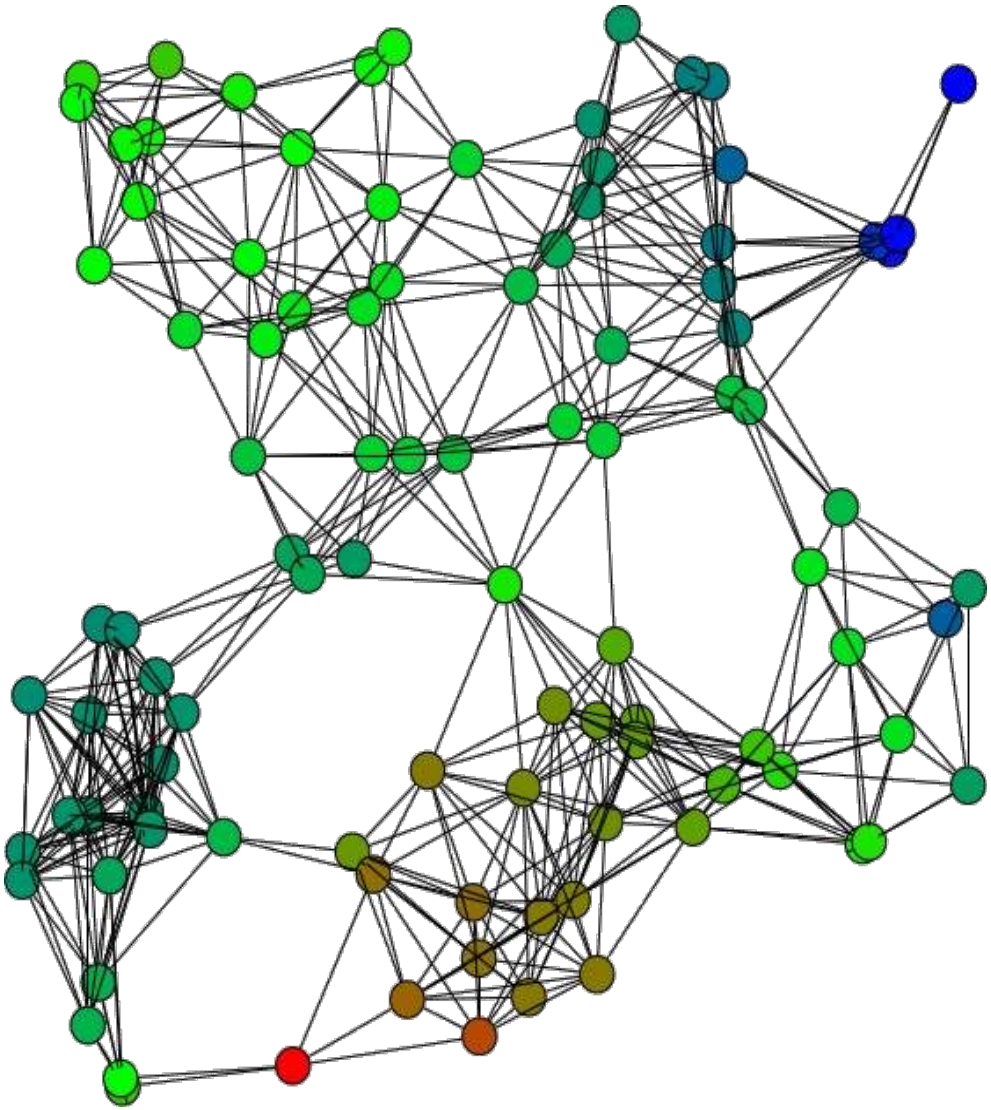


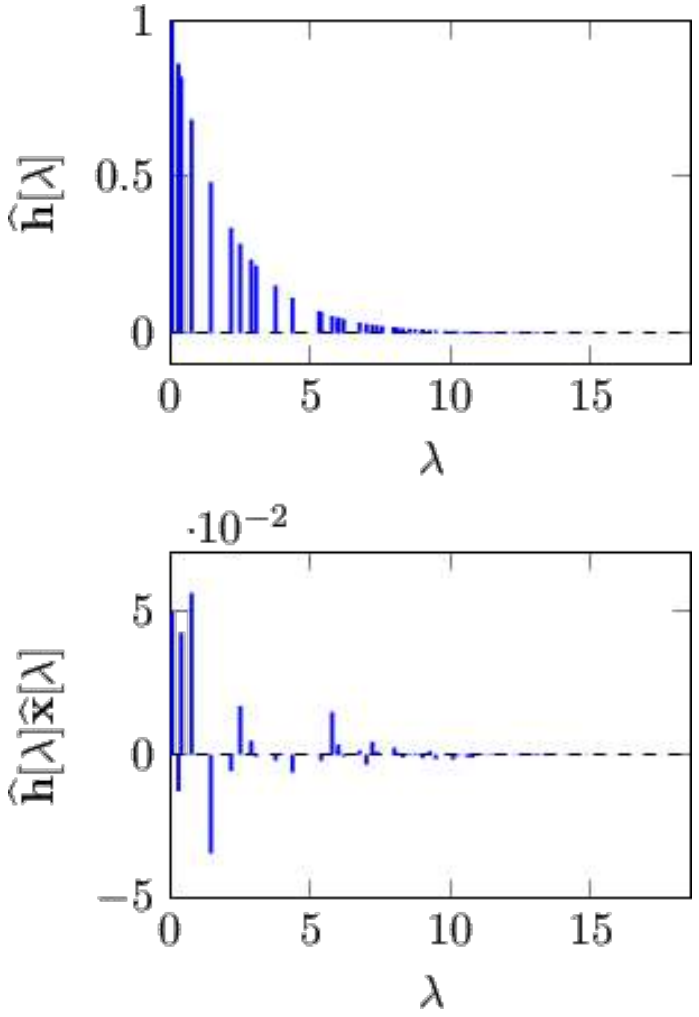
Fourier transform: projection on eigenvectors of the Laplacian matrix

Methods & notions — The origins of GSP

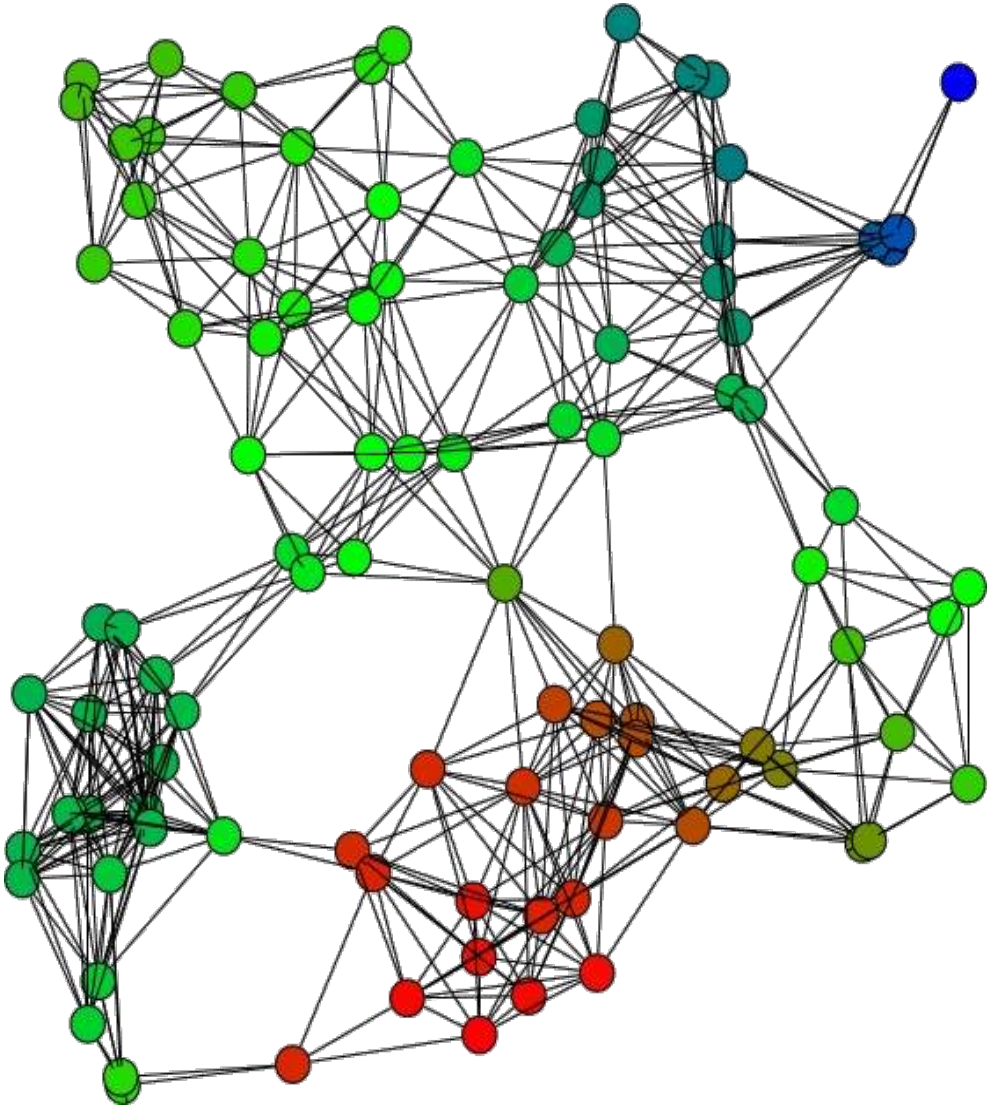


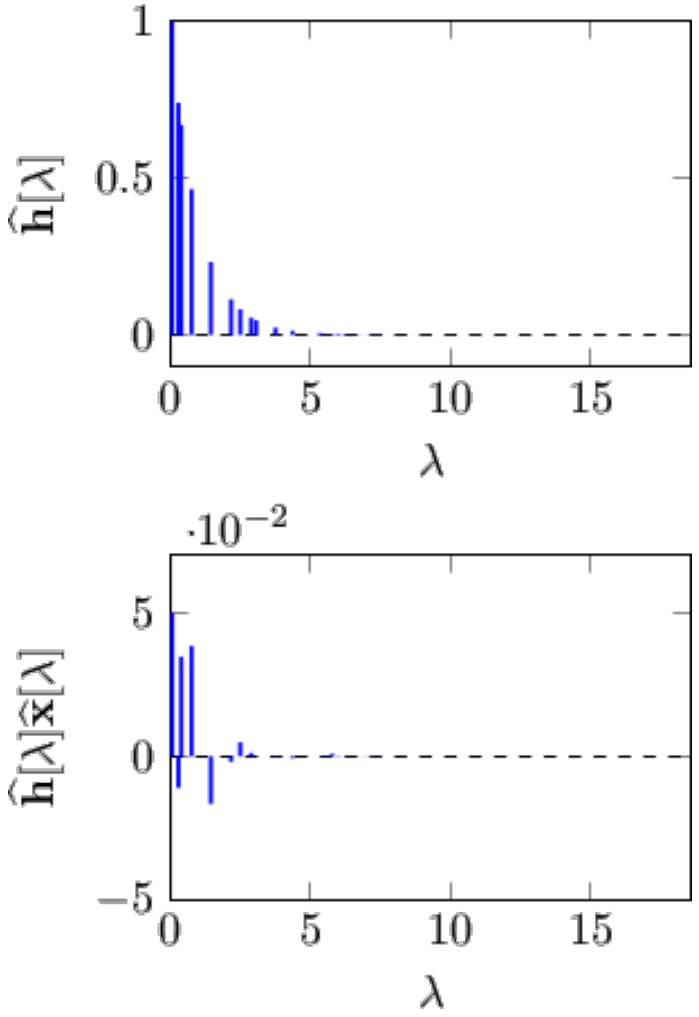
Methods & notions — The origins of GSP



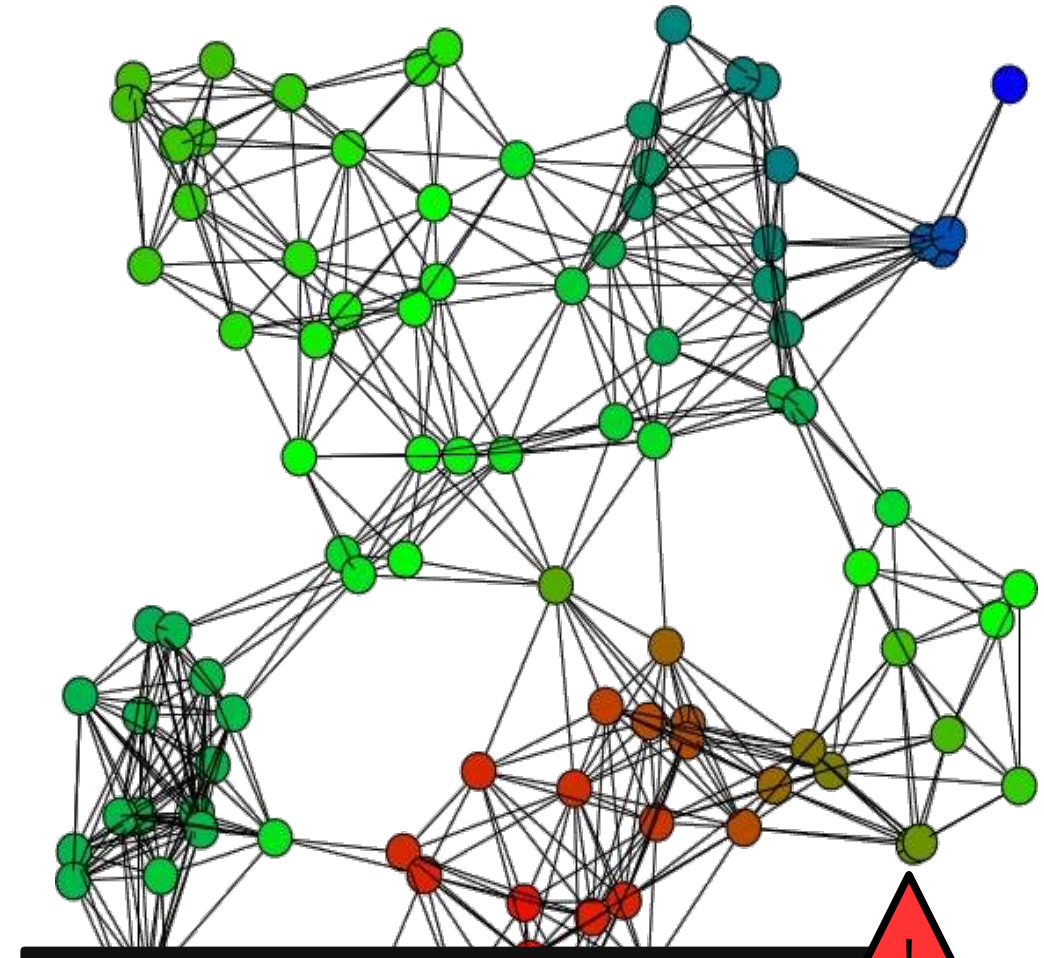


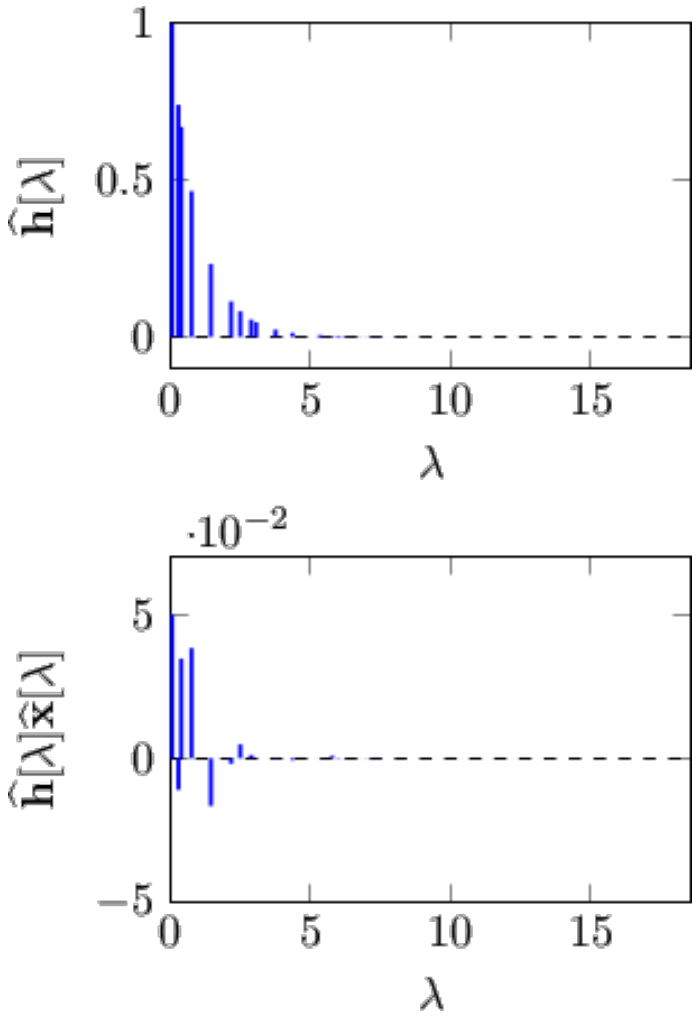
Methods & notions — The origins of GSP





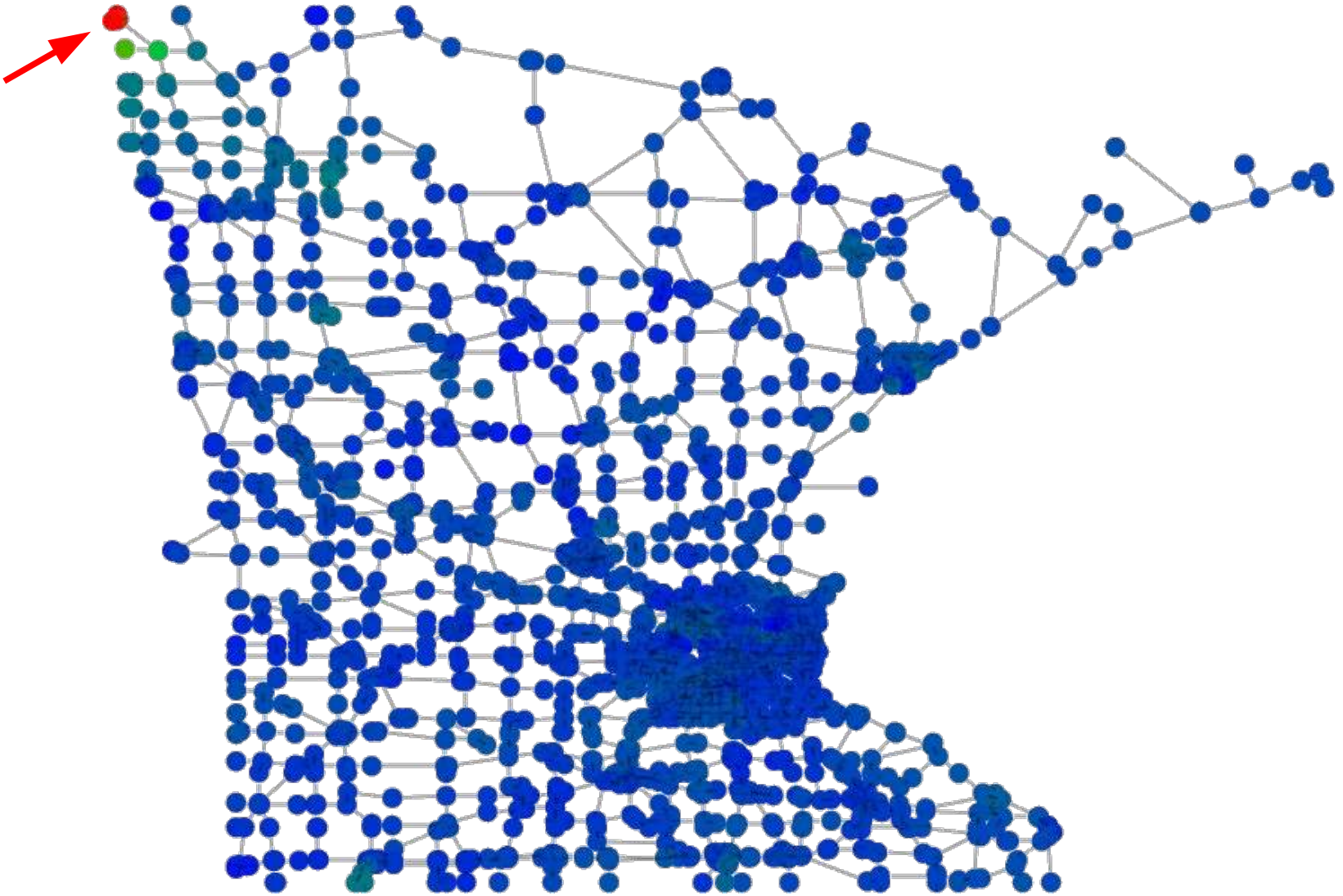
Methods & notions — The origins of GSP



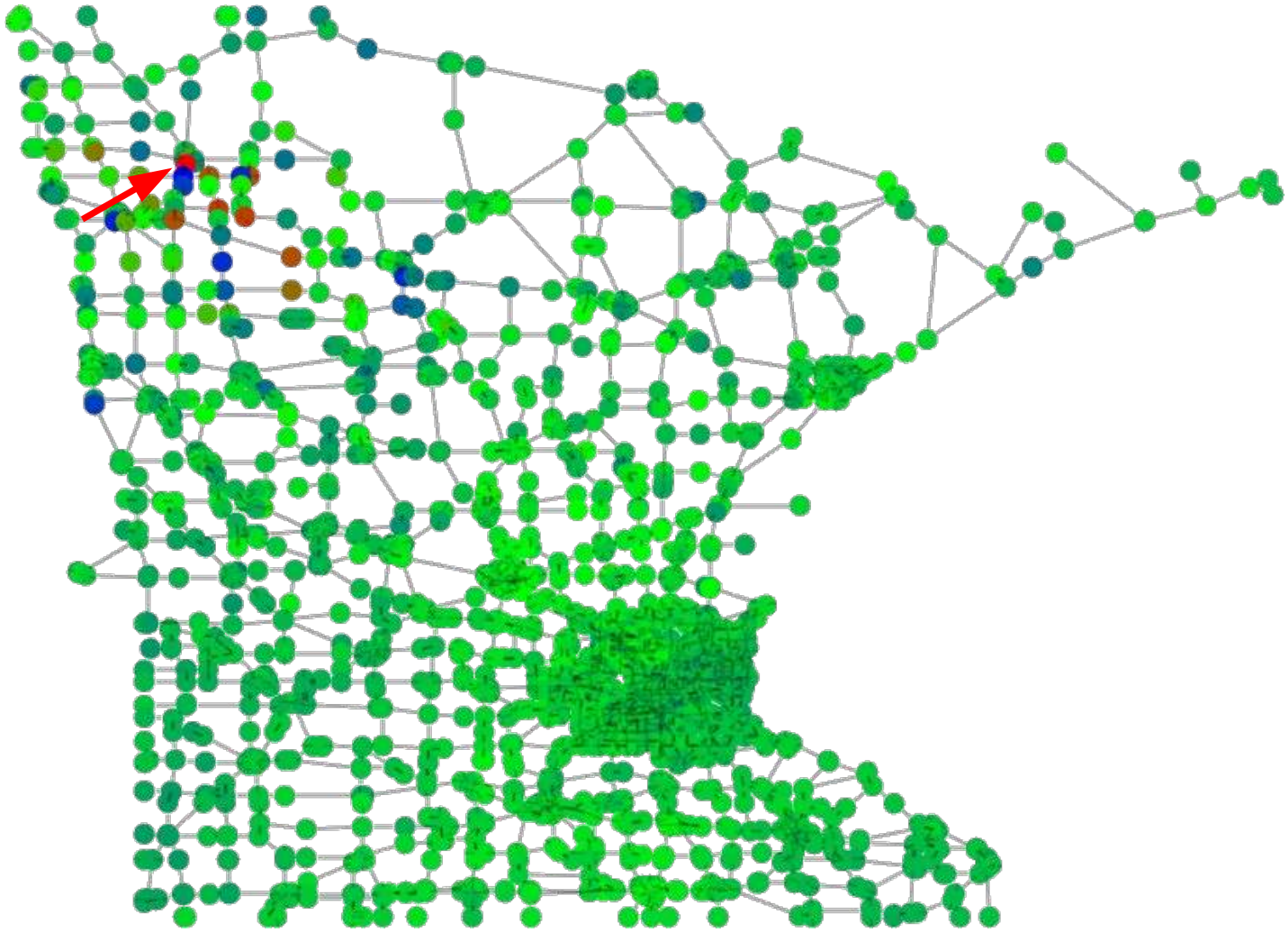


Eigenvalues are analogous to frequencies
This generalizes to all graphs

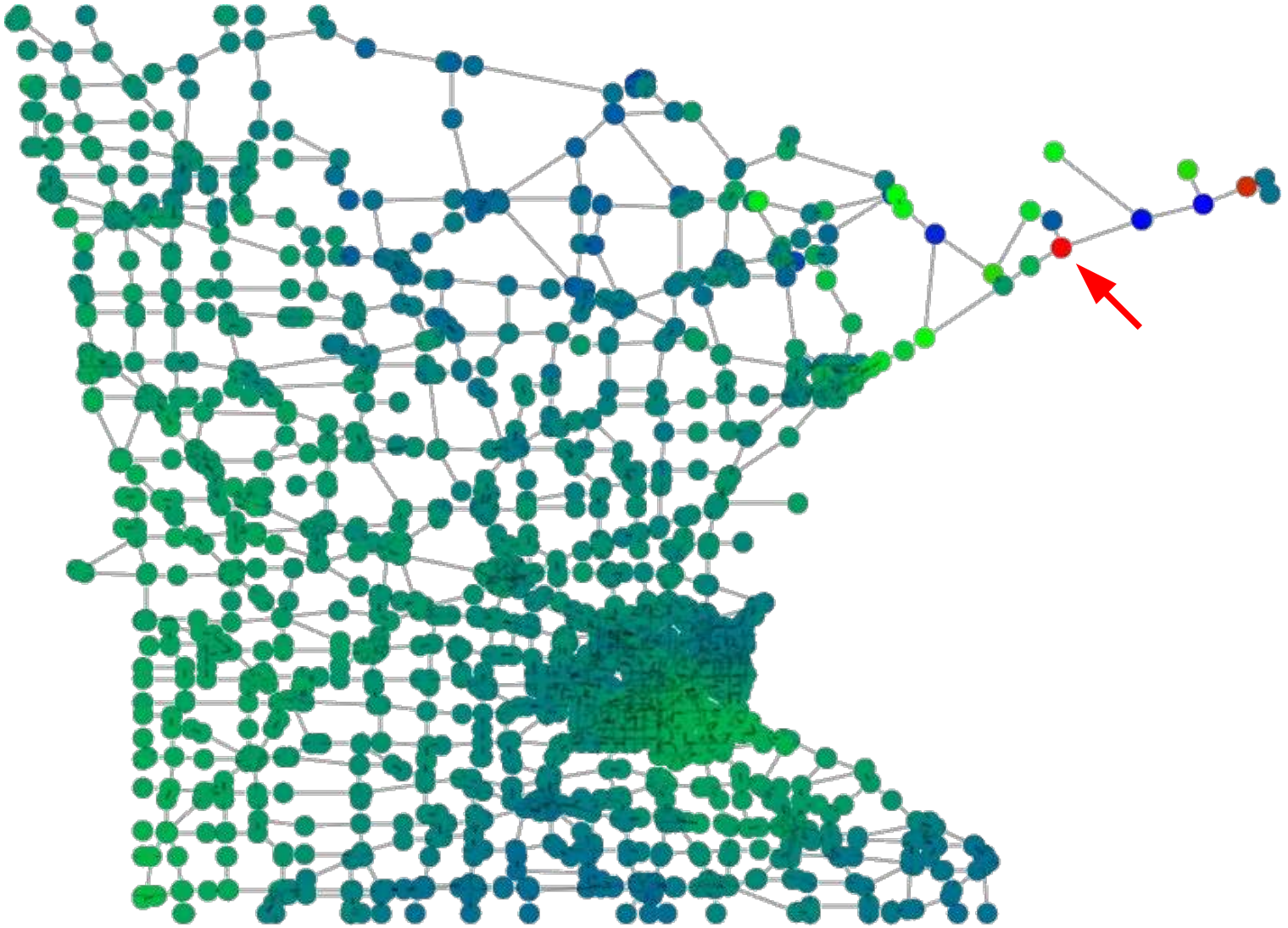
Methods & notions — Translation in the GSP framework



Methods & notions — Translation in the GSP framework



Methods & notions — Translation in the GSP framework



Methods & notions — Two families of methods

λ

Bruna *et al.* — Spectral networks and locally connected networks on graphs (2013)

Henaff *et al.* — Deep convolutional networks on graph-structured data (2015)

Defferrard *et al.* — Convolutional neural networks on graphs with fast localized spectral filtering (2016)

Atwood *et al.* — Search-convolutional neural networks (2015)

Puy *et al.* — Unifying local and non-local signal processing with graph CNNs (2017)

Hechtlinger *et al.* — A Generalization of convolutional neural networks to graph-structured data (2017)

Vialatte *et al.* — Learning local receptive fields and their weight sharing scheme on graphs (2017)

$\mathcal{G}$

Outline

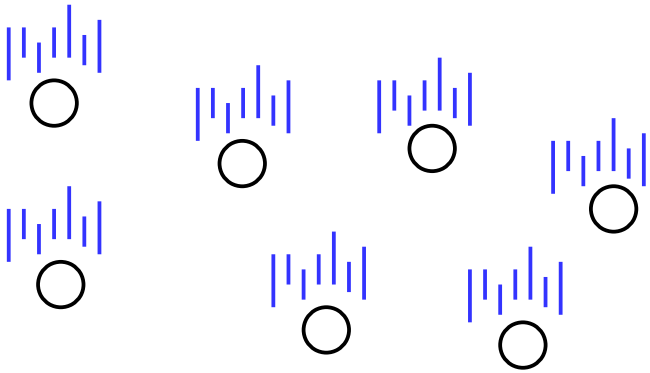
Existing methods & required notions

The method we propose

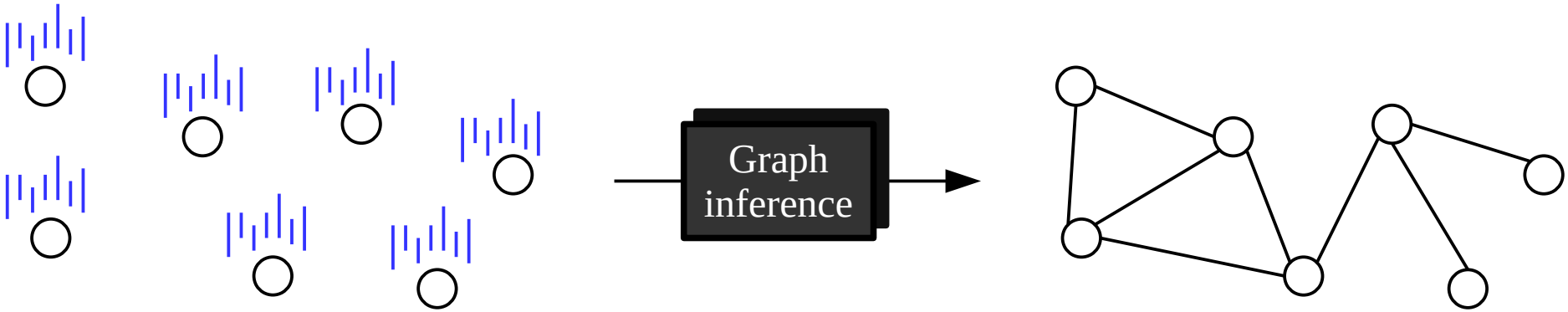
— Global idea of the method —
— Graph inference from data —
— Identification of translations —

Results

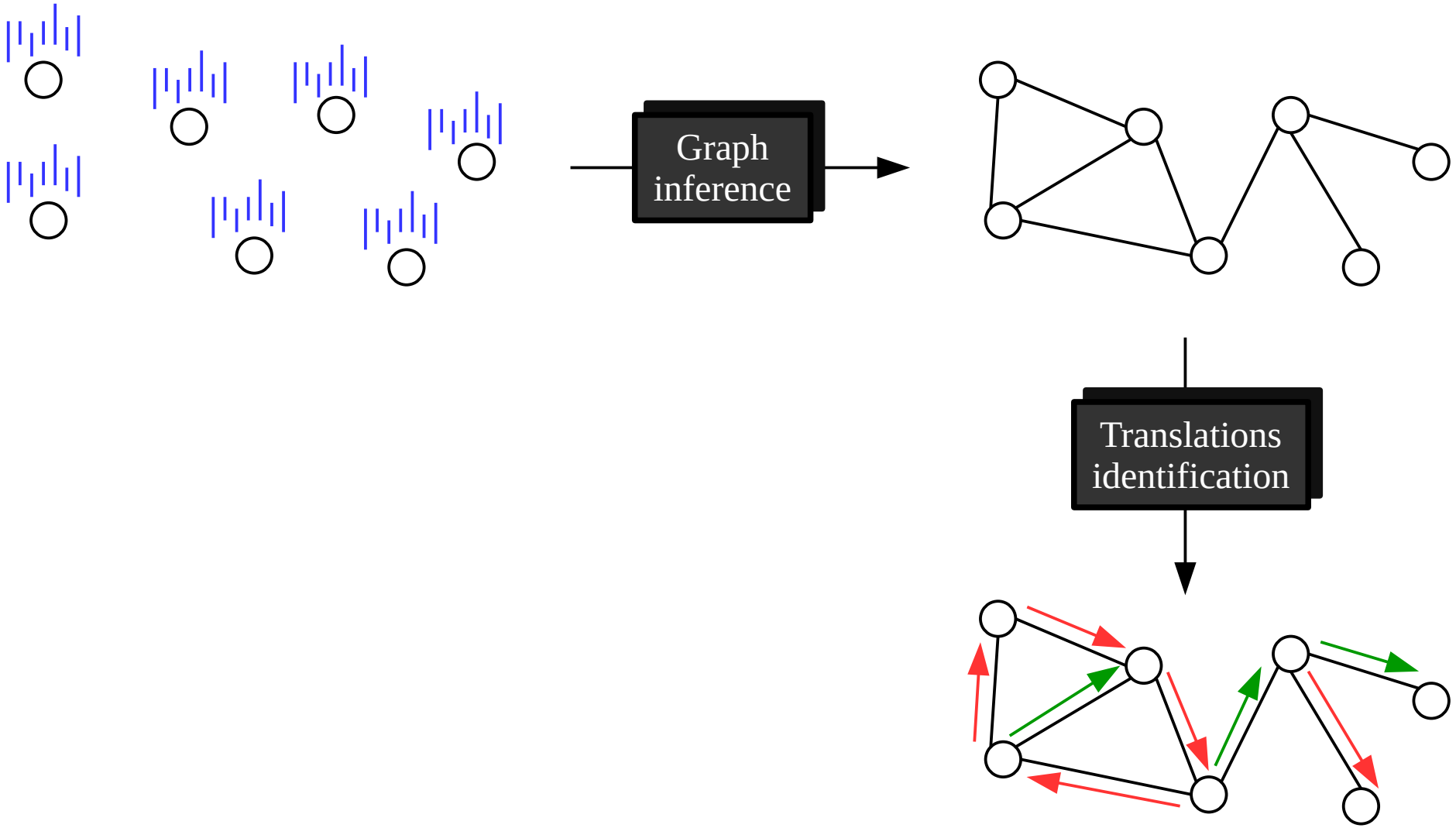
Our method — Global idea — From signals to CNNs



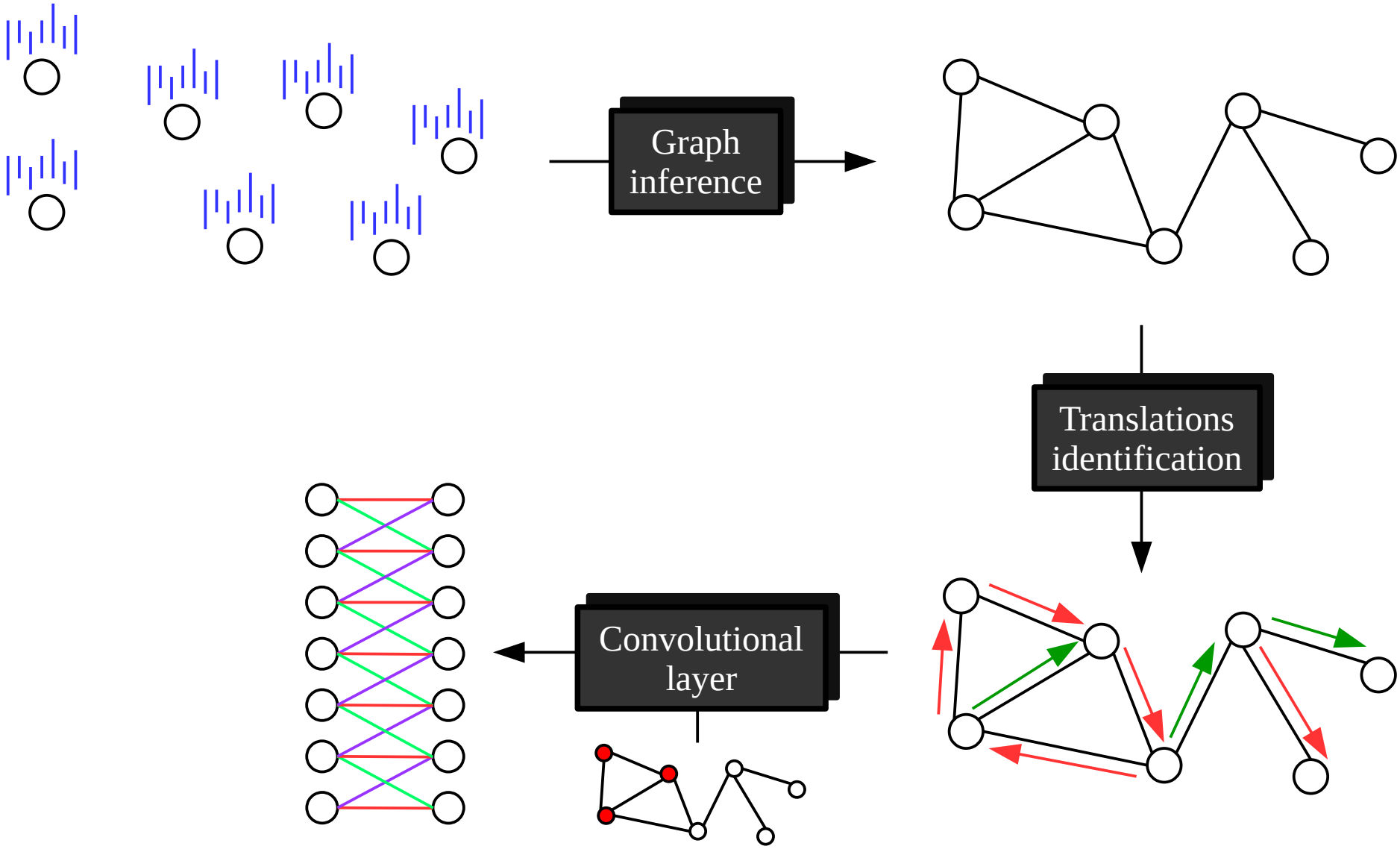
Our method — Global idea — From signals to CNNs



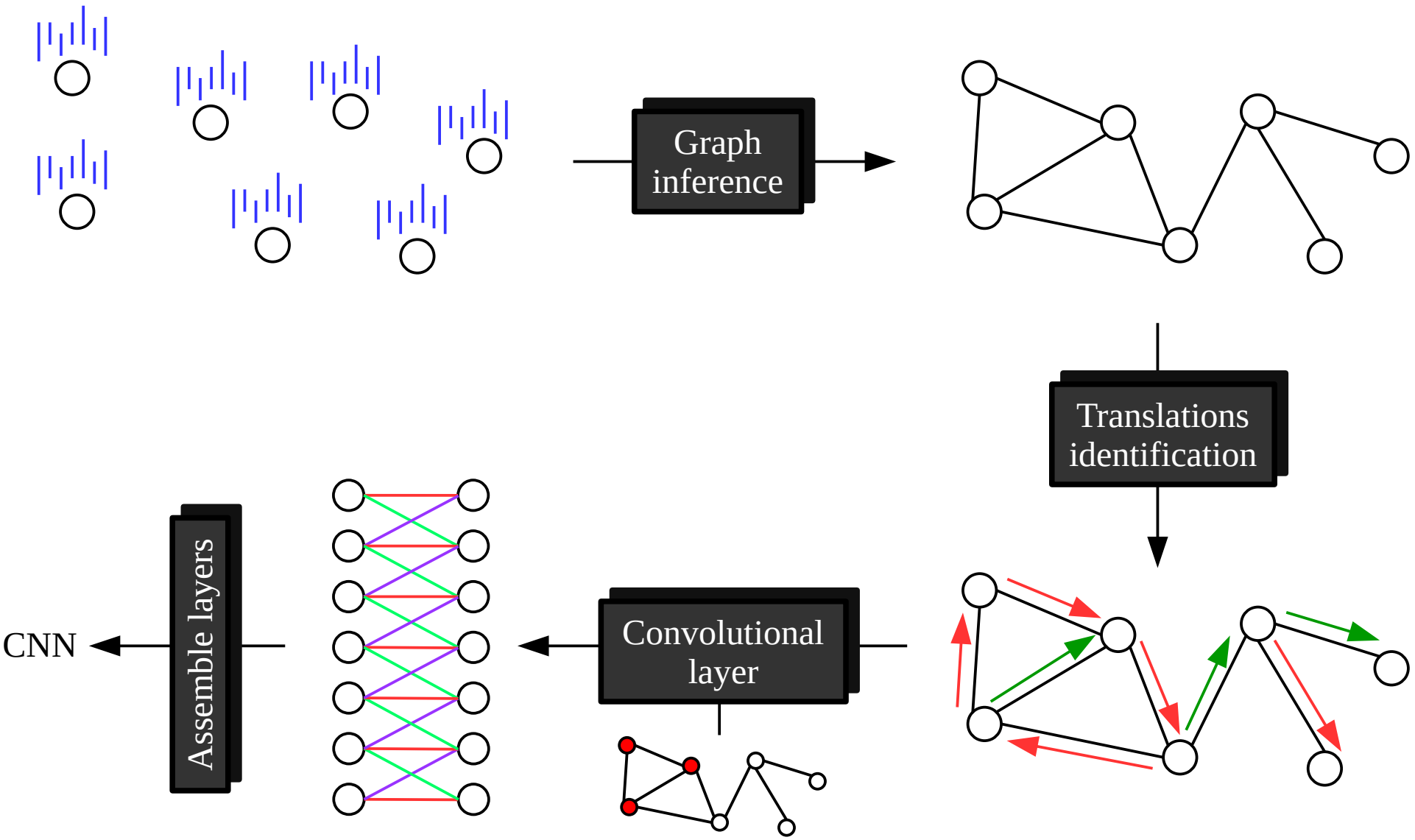
Our method — Global idea — From signals to CNNs



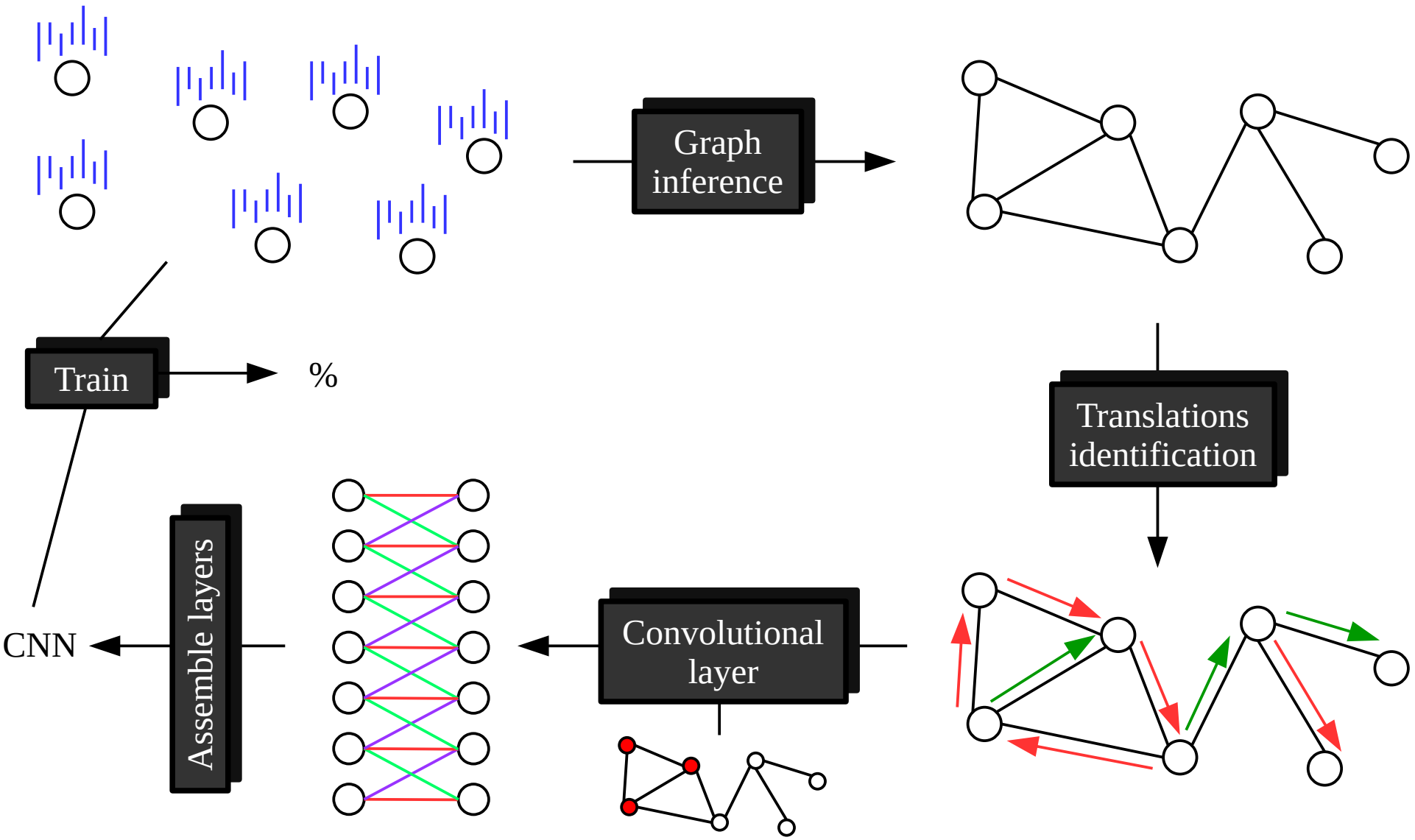
Our method — Global idea — From signals to CNNs



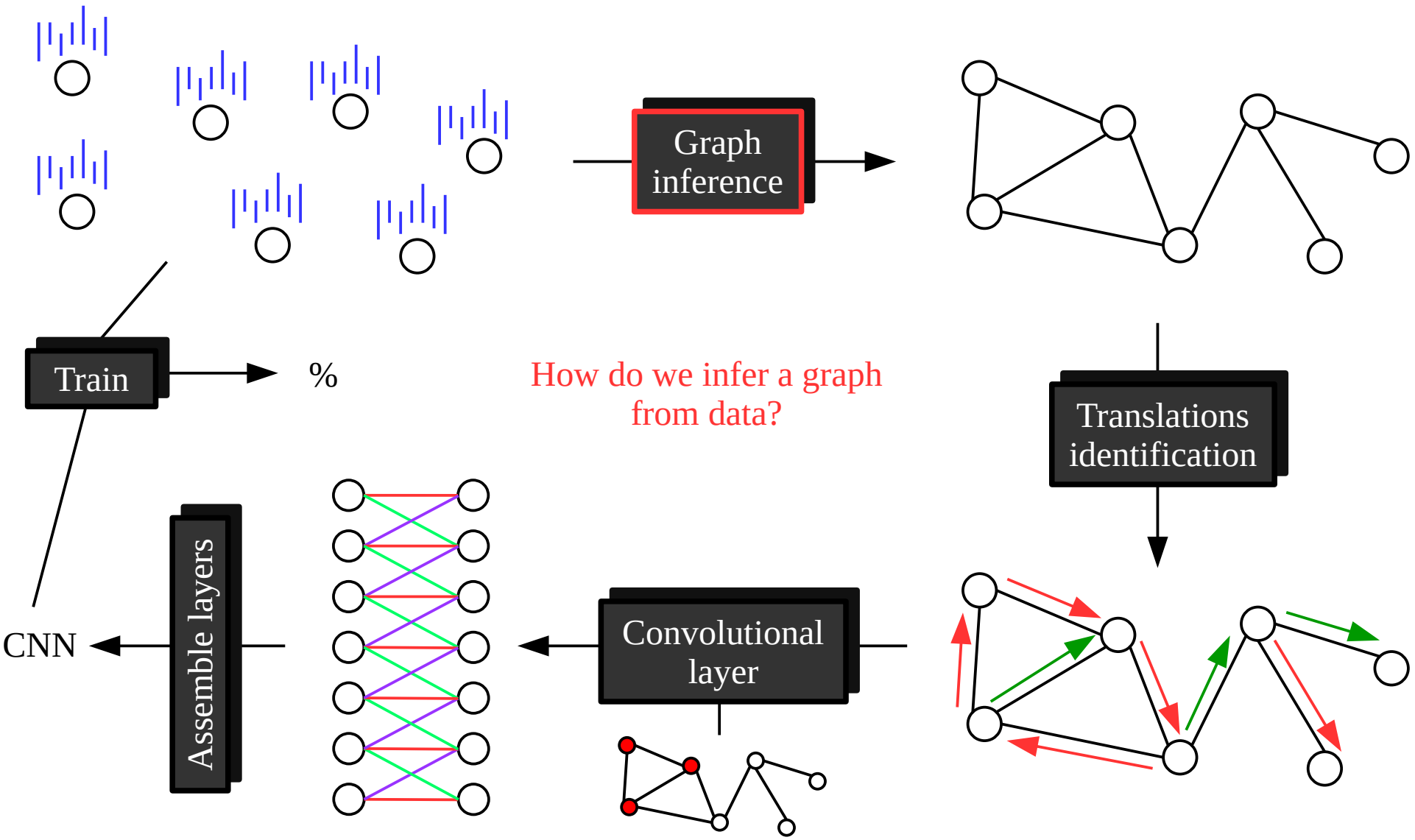
Our method — Global idea — From signals to CNNs



Our method — Global idea — From signals to CNNs



Our method — Graph inference



$$2^{\frac{N(N-1)}{2}}$$

Our method — Graph inference — An ill-posed problem

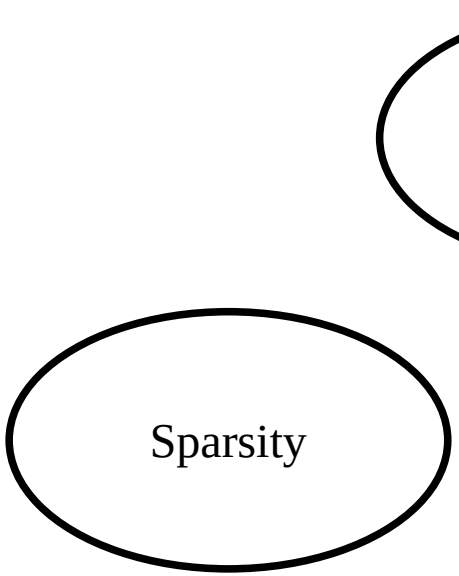
Sparsity

$$2^{\frac{N(N-1)}{2}}$$

Friedman *et al.* – Sparse inverse covariance estimation with the graphical lasso (2008)

Pavez *et al.* – Generalized Laplacian precision matrix estimation for graph signal processing (2006)

Our method — Graph inference — An ill-posed problem



$$2^{\frac{N(N-1)}{2}}$$

Lake *et al.* — Discovering structure by Learning sparse graphs (2010)

Kalofolias — How to learn a graph from smooth signals (2016)

Our method — Graph inference — An ill-posed problem

Smoothness

Stationarity / Diffusion

Sparsity

$$2^{\frac{N(N-1)}{2}}$$

Pasdeloup *et al.* – Graph reconstruction from the observation of diffused signals (2015)

Pasdeloup *et al.* – Characterization and inference of graph diffusion processes from observations of stationary signals (2017)

Segarra *et al.* – Network topology identification from spectral templates (2016)

Our method — Graph inference — An ill-posed problem

Smoothness

Stationarity / Diffusion

Sparsity

$$2^{\frac{N(N-1)}{2}}$$

Time series

Mei *et al.* — Signal processing on graphs: Estimating the structure of a graph (2015)

Shen *et al.* — Topology inference of directed graphs using nonlinear structural vector autoregressive models (2017)

Our method — Graph inference — An ill-posed problem

Smoothness

Stationarity / Diffusion

Sparsity

$$2^{\frac{N(N-1)}{2}}$$

Time series

Spike models

Thanou *et al.* — Learning heat diffusion graphs (2016)

Petric Maretic *et al.* — Graph learning under sparsity priors (2017)

Our method — Graph inference — An ill-posed problem

Smoothness

Stationarity / Diffusion

Sparsity

$$2^{\frac{N(N-1)}{2}}$$

Time series

Spike models

Smoothness $\Leftrightarrow$ Locality
Stationarity $\Leftrightarrow$ Weight-sharing

Our method — Graph inference — A model of stationary signals

Matrix representing the graph (GSO)

Observed signal

White noise

$$\mathbf{x} = \sum_{k \in \mathcal{K}} \alpha_k \mathbf{T}^k \mathbf{y}$$

Matrix representing the graph (GSO)

Observed signal

White noise

$$\mathbf{x} = \sum_{k \in \mathcal{K}} \alpha_k \mathbf{T}^k \mathbf{y}$$

$$\begin{aligned} \boldsymbol{\Sigma} = & \quad \mathbb{E}[\mathbf{x}\mathbf{x}^\top] \\ = & \quad \mathbb{E}\left[\sum_{k \in \mathcal{K}} \beta_k \mathbf{T}^k \mathbf{y}\mathbf{y}^\top\right] \\ \propto & \quad \sum_{k \in \mathcal{K}} \beta_k \mathbf{T}^k \end{aligned}$$

Matrix representing the graph (GSO)

Observed signal

White noise

$$\mathbf{x} = \sum_{k \in \mathcal{K}} \alpha_k \mathbf{T}^k \mathbf{y}$$

$$\begin{aligned} \mathbf{\Sigma} = & \quad \mathbb{E}[\mathbf{x}\mathbf{x}^\top] \\ = & \quad \mathbb{E}\left[\sum_{k \in \mathcal{K}} \beta_k \mathbf{T}^k \mathbf{y}\mathbf{y}^\top\right] \\ \propto & \quad \sum_{k \in \mathcal{K}} \beta_k \mathbf{T}^k \end{aligned}$$

Same eigenvectors for Σ and $\mathbf{T}$
Hard to obtain in practice

$$\mathbf{T} = \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^\top$$

$$\mathbf{T} = \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^\top$$

$$\forall i, j : \mathbf{T}[i, j] = \alpha\lambda_1 + \beta\lambda_2 + \cdots + \gamma\lambda_N$$

$$\mathbf{T} = \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^\top$$

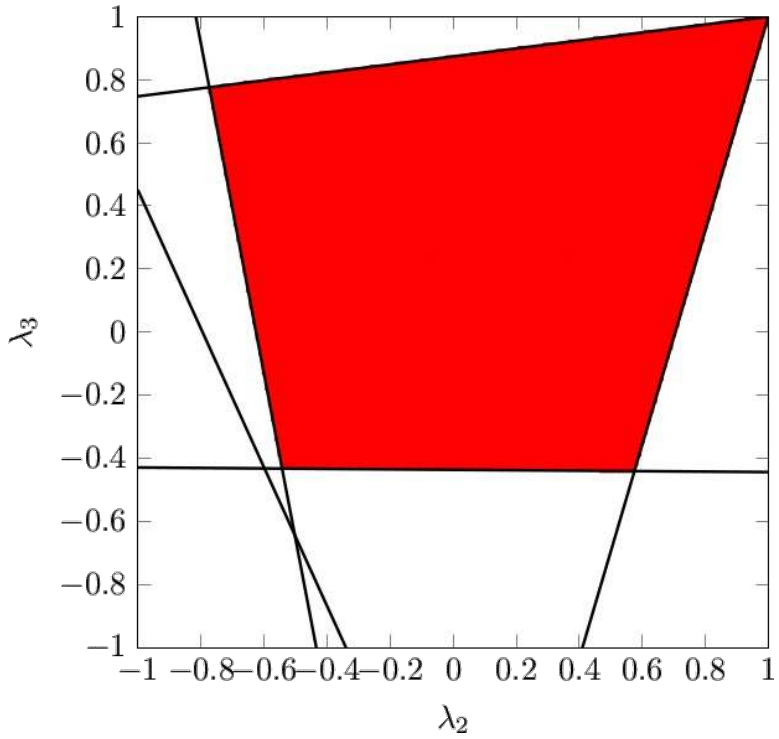
$$\forall i, j : \mathbf{T}[i, j] = \alpha\lambda_1 + \beta\lambda_2 + \cdots + \gamma\lambda_N$$

$$\forall i, j : \mathbf{T}[i, j] \geq 0$$

$$\mathbf{T} = \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^\top$$

$$\forall i, j : \mathbf{T}[i, j] = \alpha\lambda_1 + \beta\lambda_2 + \cdots + \gamma\lambda_N$$

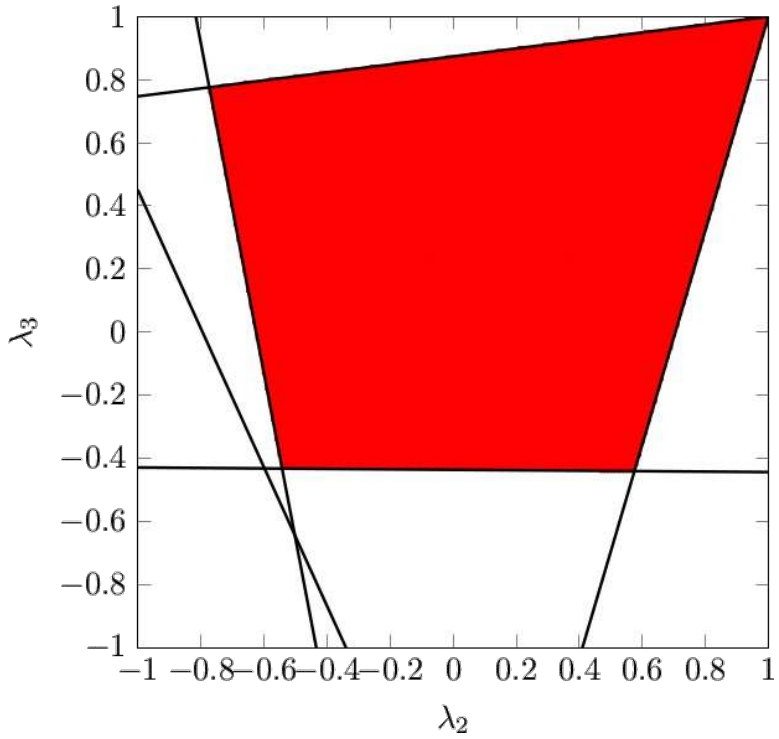
$$\forall i, j : \mathbf{T}[i, j] \geq 0$$



$$\mathbf{T} = \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^\top$$

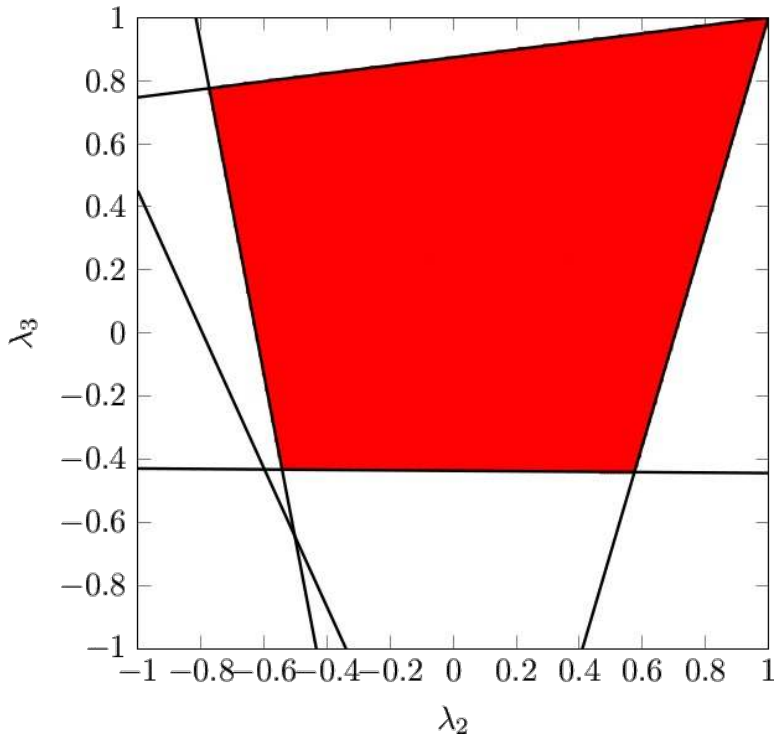
$$\forall i, j : \mathbf{T}[i, j] = \alpha\lambda_1 + \beta\lambda_2 + \cdots + \gamma\lambda_N$$

$$\forall i, j : \mathbf{T}[i, j] \geq 0$$



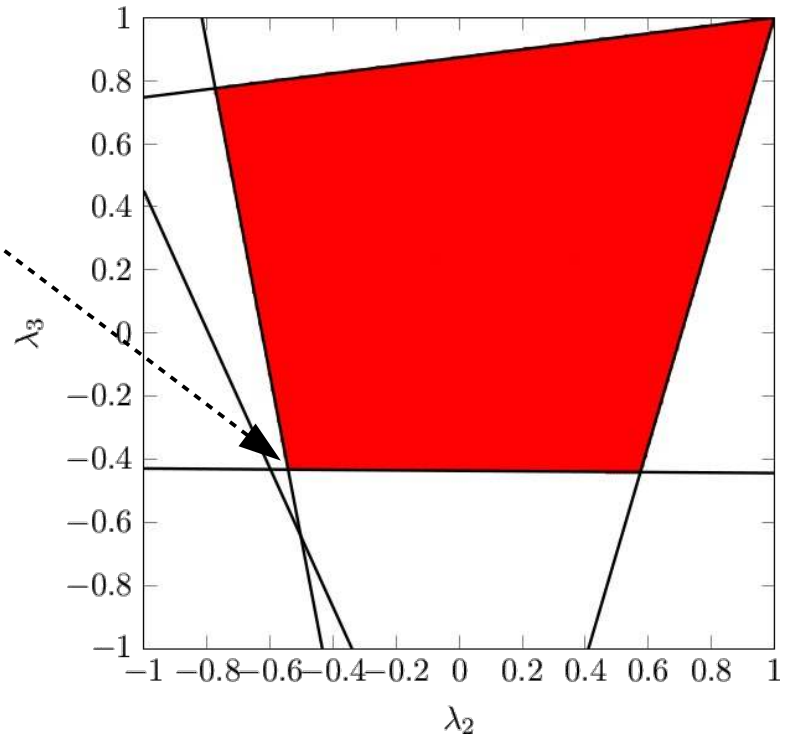
Admissible matrices lie in a convex set
Need to select a vector of eigenvalues

Our method — Graph inference — Eigenvalues selection



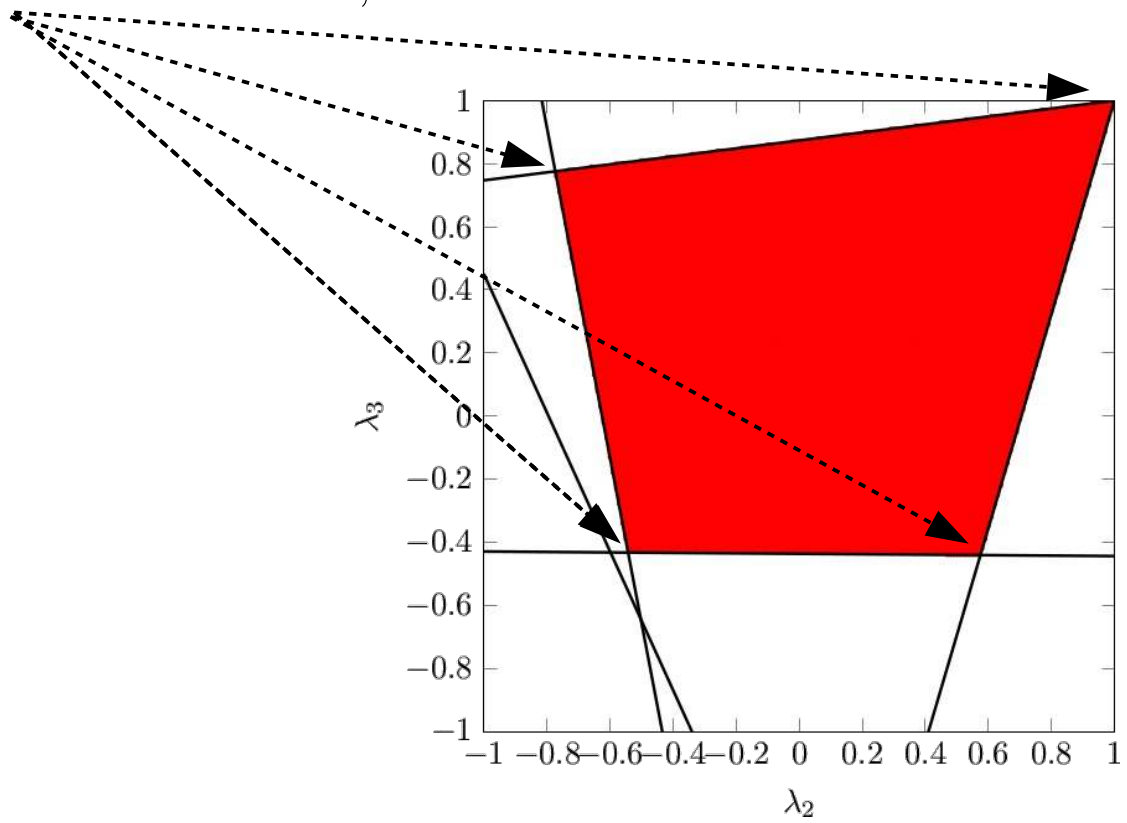
Our method — Graph inference — Eigenvalues selection

$$\arg \min_{\lambda_1,\ldots,\lambda_N} \sum_{i=1}^{N} \lambda_i$$



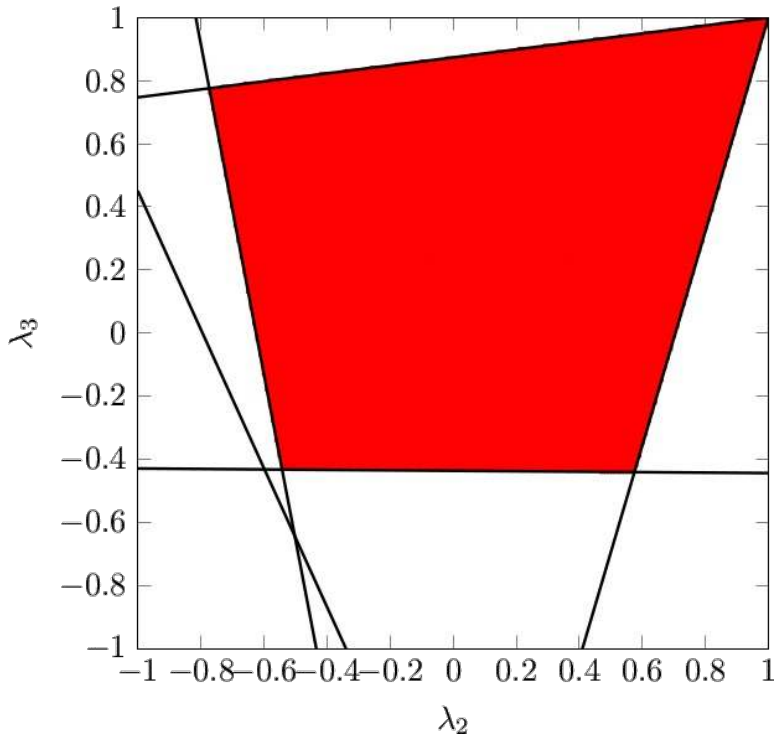
Our method — Graph inference — Eigenvalues selection

$$\arg \min_{\lambda_1,\ldots,\lambda_N} \left\| \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^\top \right\|_{1,1}$$



Our method — Graph inference — Eigenvalues selection

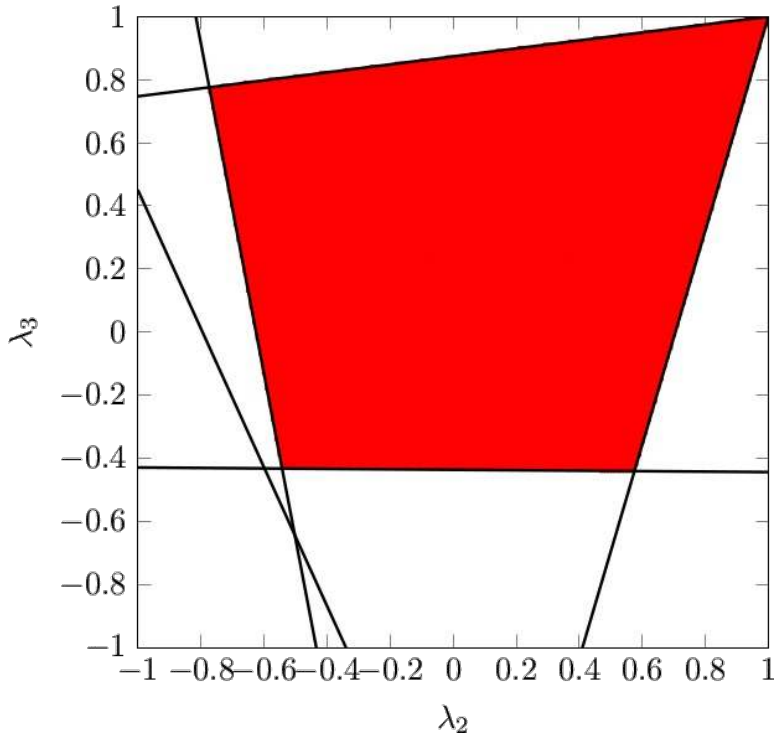
$$\mathbf{T} = \mathcal{X}_{\mathbf{T}} \begin{bmatrix} \lambda_{\mathbf{T}1} & 0 & 0 \\ 0 & \lambda_{\mathbf{T}2} & 0 \\ 0 & 0 & \lambda_{\mathbf{T}3} \end{bmatrix} \mathcal{X}_{\mathbf{T}}^{\top}$$



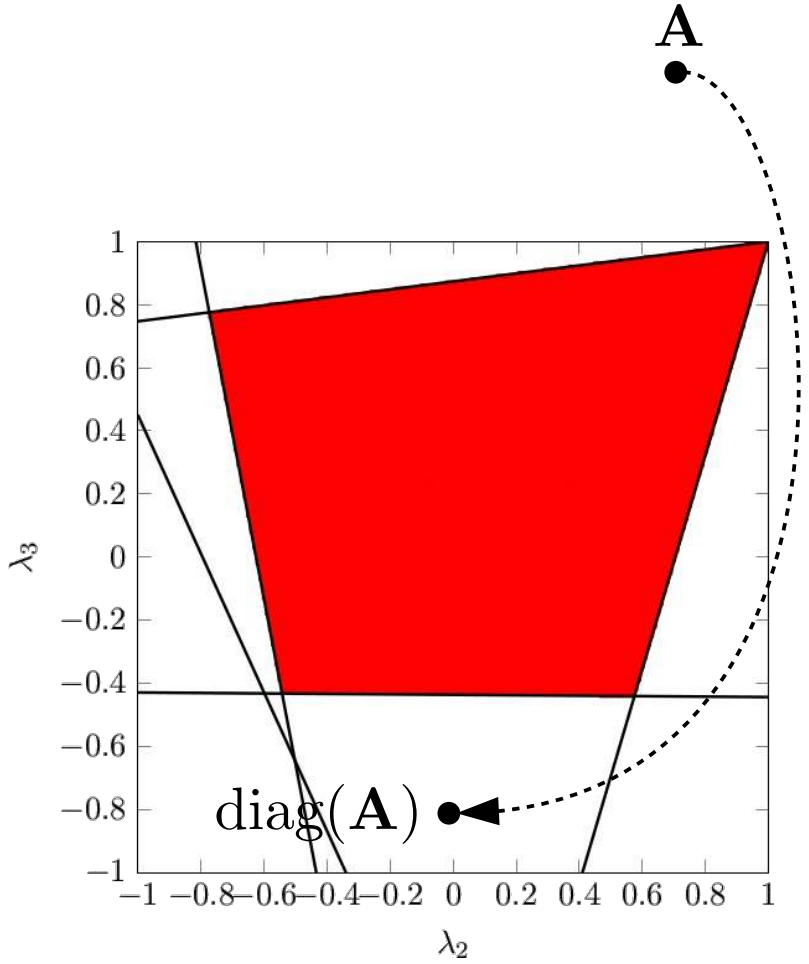
Our method — Graph inference — Eigenvalues selection

$$\begin{aligned}\mathbf{T} &= \mathcal{X}_{\mathbf{T}} \begin{bmatrix} \lambda_{\mathbf{T}1} & 0 & 0 \\ 0 & \lambda_{\mathbf{T}2} & 0 \\ 0 & 0 & \lambda_{\mathbf{T}3} \end{bmatrix} \mathcal{X}_{\mathbf{T}}^{\top} \\ &= \mathcal{X}\mathbf{A}\mathcal{X}^{\top}\end{aligned}$$

$\mathbf{A}$

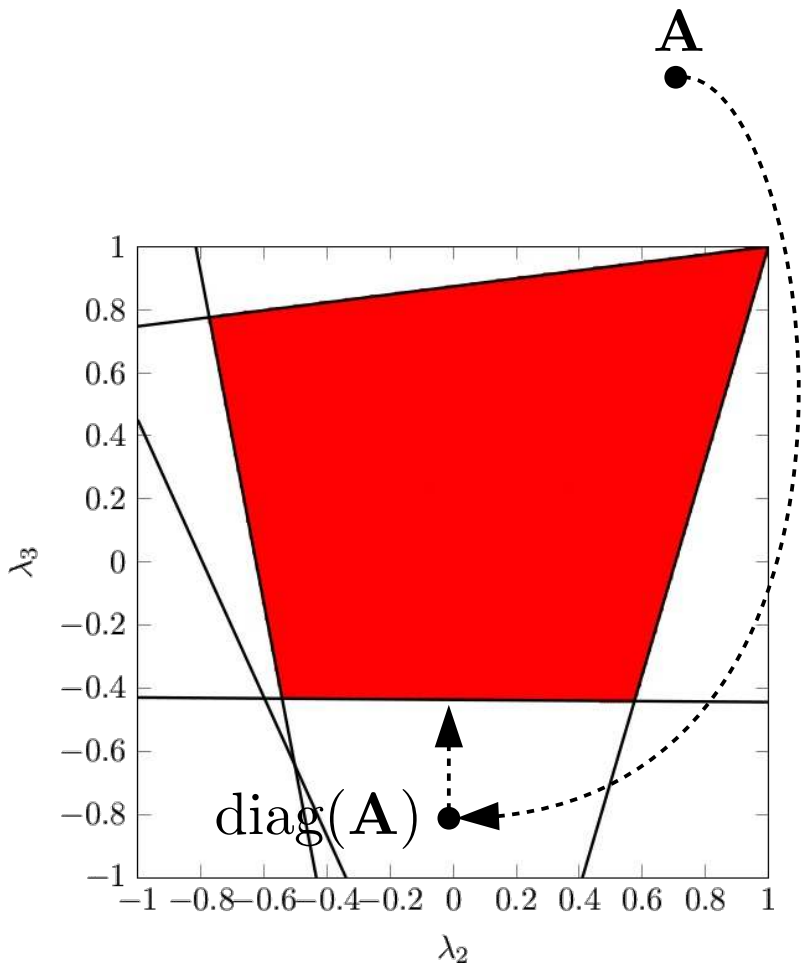


$$\begin{aligned}\mathbf{T} = & \quad \mathcal{X}_{\mathbf{T}} \begin{bmatrix} \lambda_{\mathbf{T}1} & 0 & 0 \\ 0 & \lambda_{\mathbf{T}2} & 0 \\ 0 & 0 & \lambda_{\mathbf{T}3} \end{bmatrix} \mathcal{X}_{\mathbf{T}}^{\top} \\ = & \quad \mathcal{X}\mathbf{A}\mathcal{X}^{\top} \\ \approx & \quad \mathcal{X}\mathrm{diag}(\mathbf{A})\mathcal{X}^{\top}\end{aligned}$$



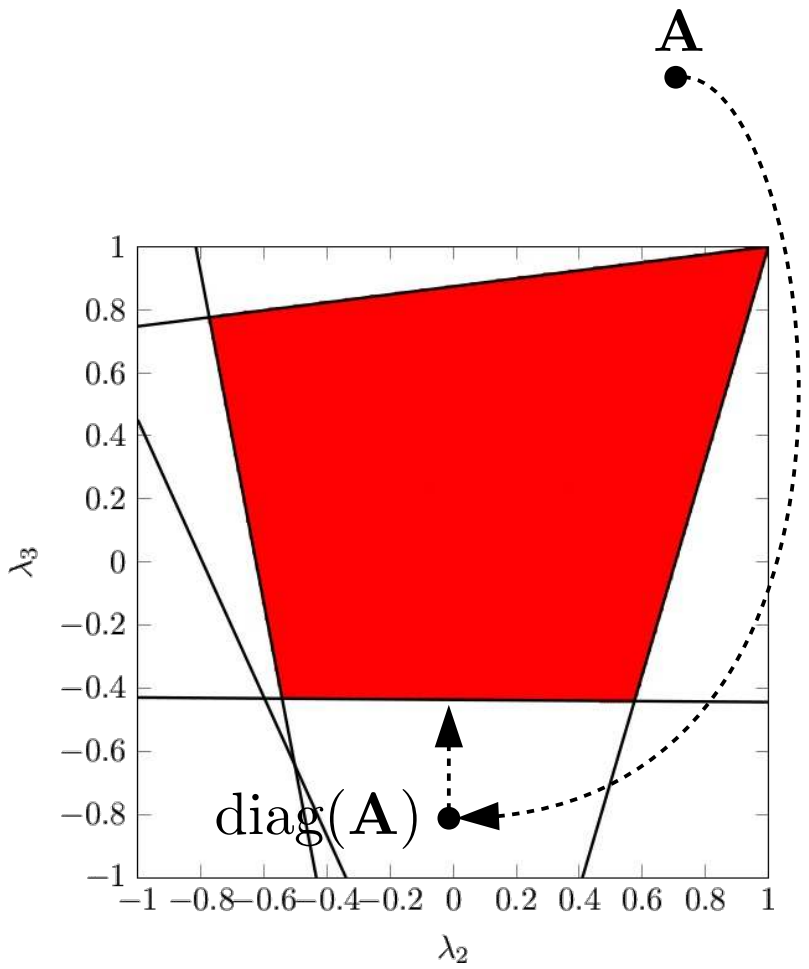
Our method — Graph inference — Eigenvalues selection

$$
\begin{aligned}
\mathbf{T} = & \quad \mathcal{X}_{\mathbf{T}} \begin{bmatrix} \lambda_{\mathbf{T}1} & 0 & 0 \\ 0 & \lambda_{\mathbf{T}2} & 0 \\ 0 & 0 & \lambda_{\mathbf{T}3} \end{bmatrix} \mathcal{X}_{\mathbf{T}}^{\top} \\
= & \quad \mathcal{X}\mathbf{A}\mathcal{X}^{\top} \\
\approx & \quad \mathcal{X}\mathrm{diag}(\mathbf{A})\mathcal{X}^{\top} \\
\approx & \quad \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^{\top}
\end{aligned}
$$



Our method — Graph inference — Eigenvalues selection

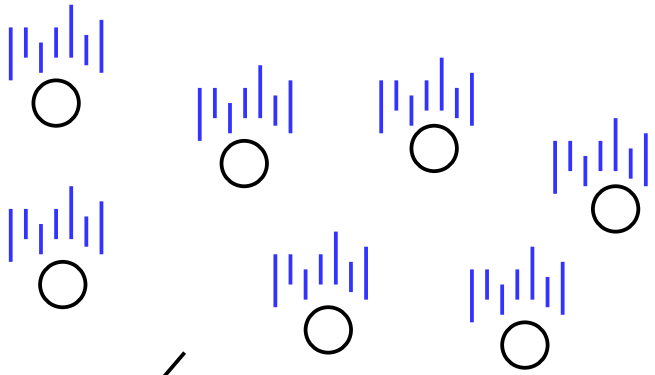
$$
\begin{aligned}
\mathbf{T} &= \mathcal{X}_{\mathbf{T}} \begin{bmatrix} \lambda_{\mathbf{T}1} & 0 & 0 \\ 0 & \lambda_{\mathbf{T}2} & 0 \\ 0 & 0 & \lambda_{\mathbf{T}3} \end{bmatrix} \mathcal{X}_{\mathbf{T}}^{\top} \\
&= \mathcal{X}\mathbf{A}\mathcal{X}^{\top} \\
&\approx \mathcal{X}\mathrm{diag}(\mathbf{A})\mathcal{X}^{\top} \\
&\approx \mathcal{X} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \mathcal{X}^{\top}
\end{aligned}
$$



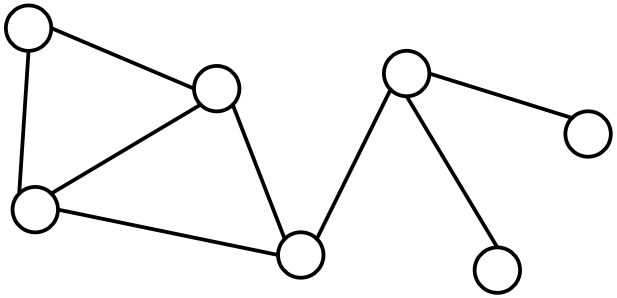
!

Countless other strategies
Depends on the desired application

Our method — Translations identification



Graph inference

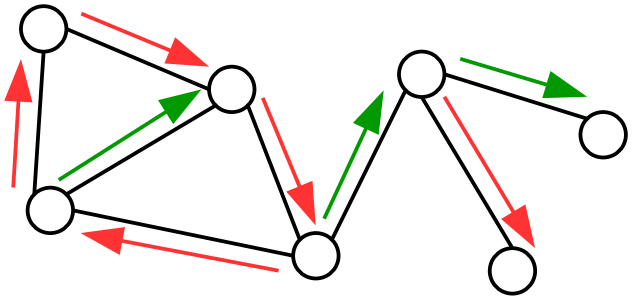


Train

%

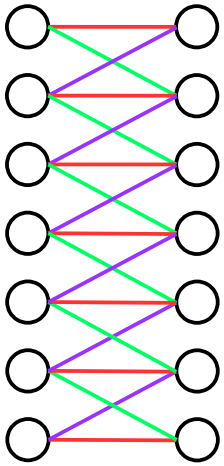
What are translations on arbitrary graphs ?

Translations identification

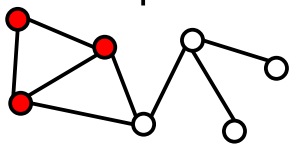


CNN

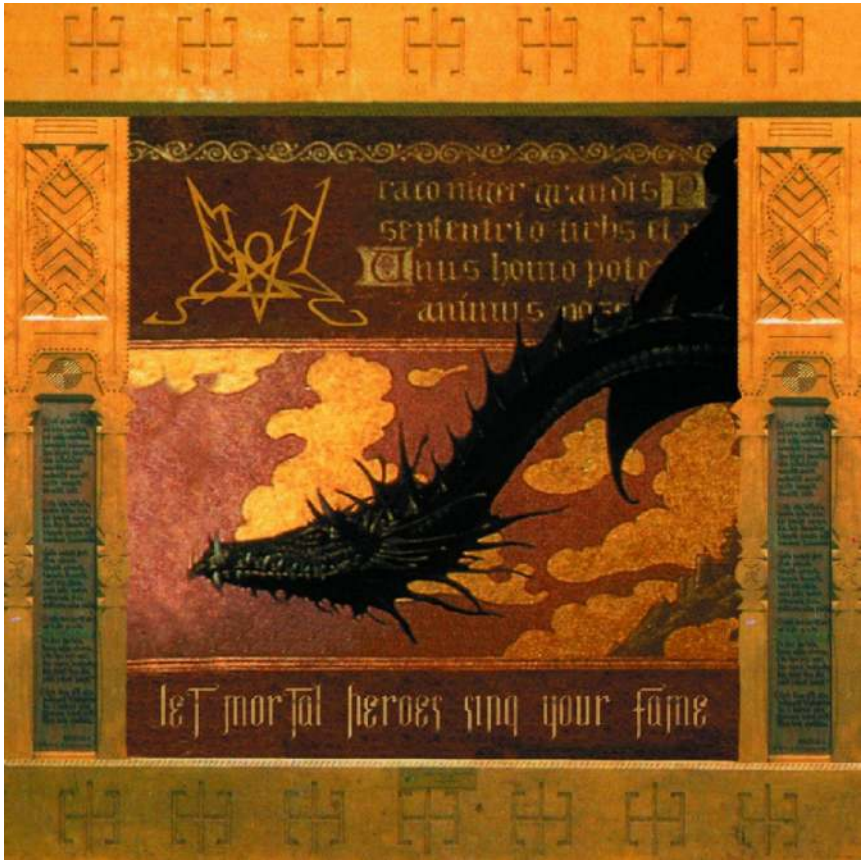
Assemble layers



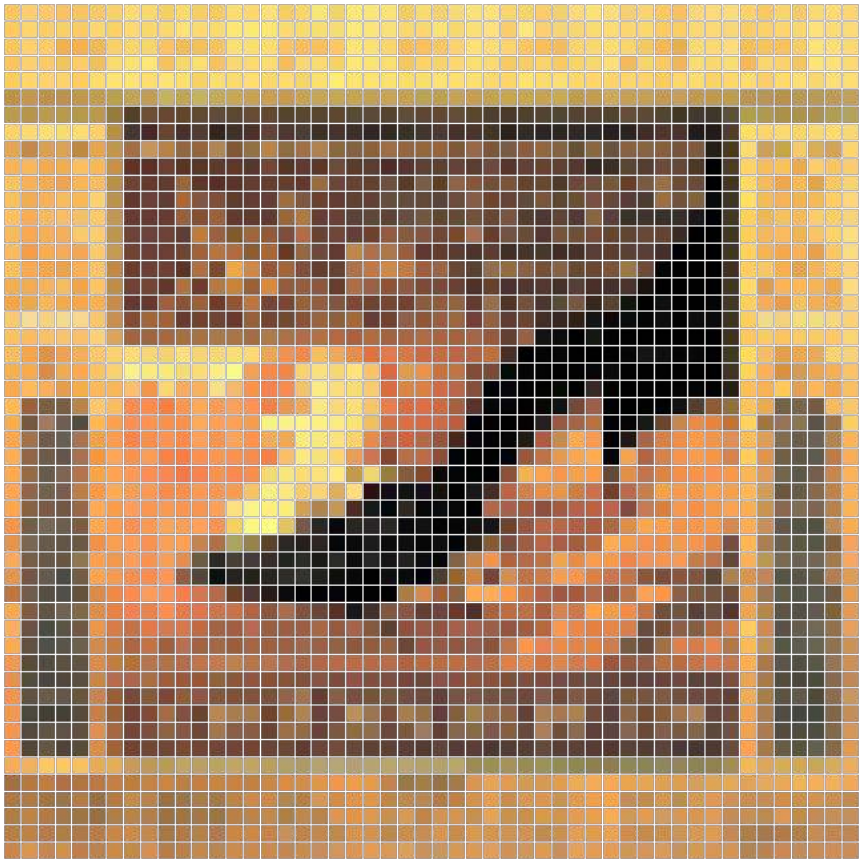
Convolutional layer



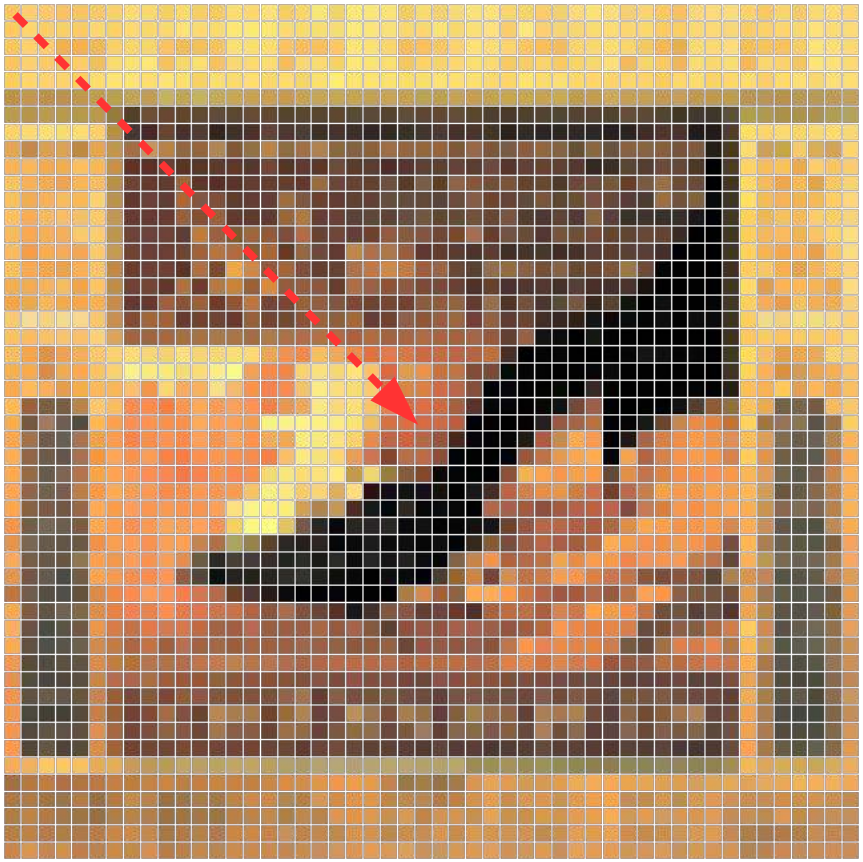
Our method — Translations identification — The case of images

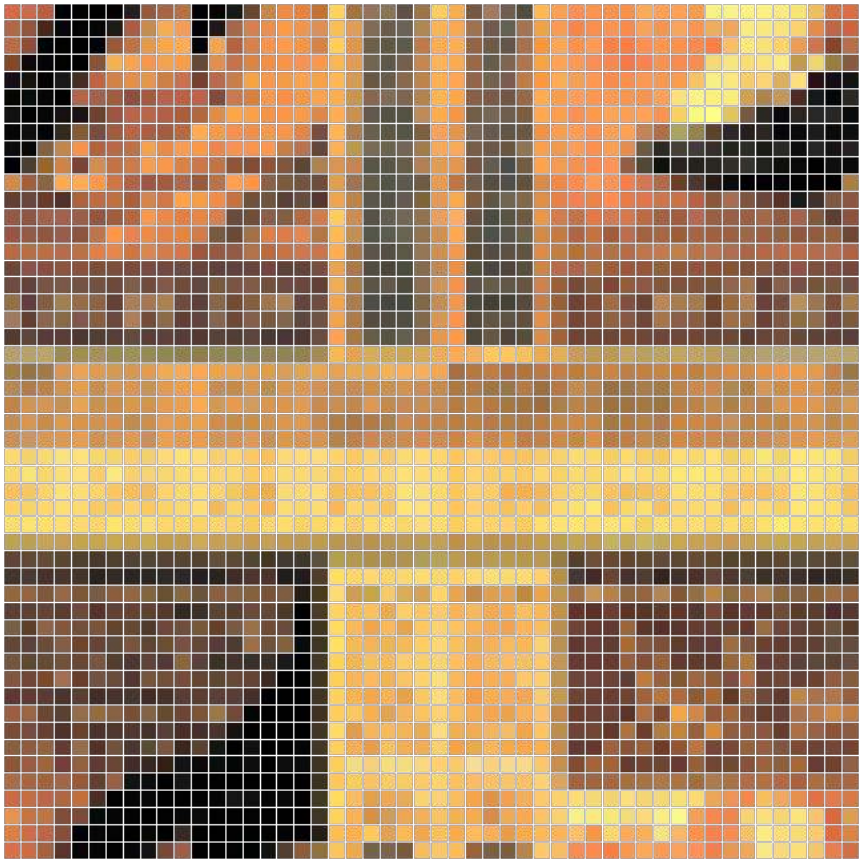


Our method — Translations identification — The case of images

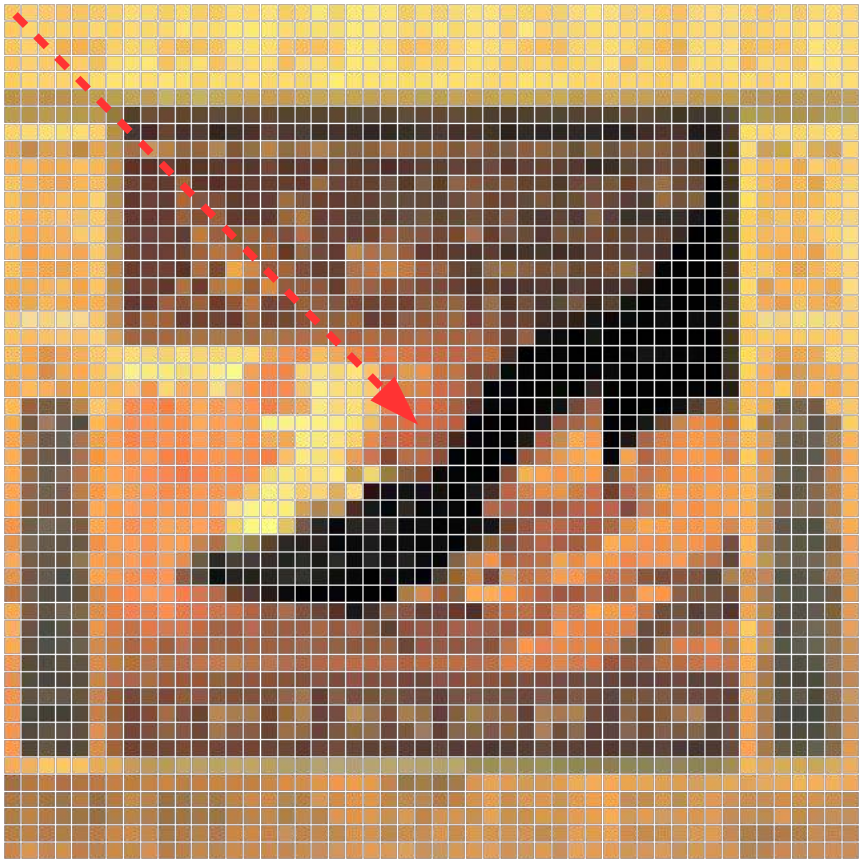


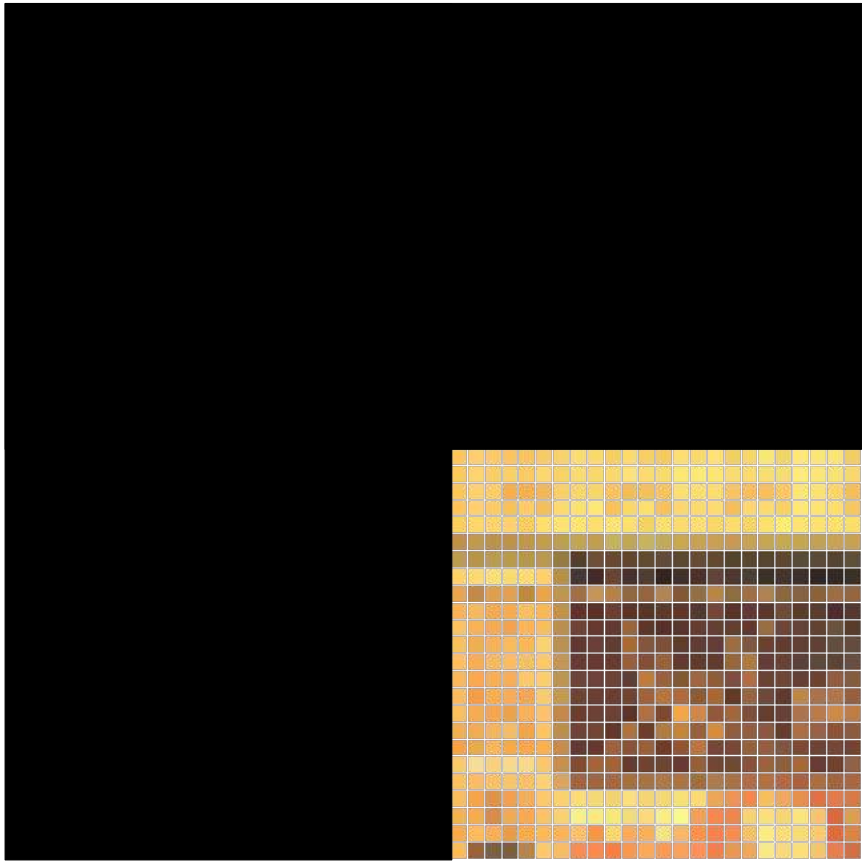
Our method — Translations identification — The case of images



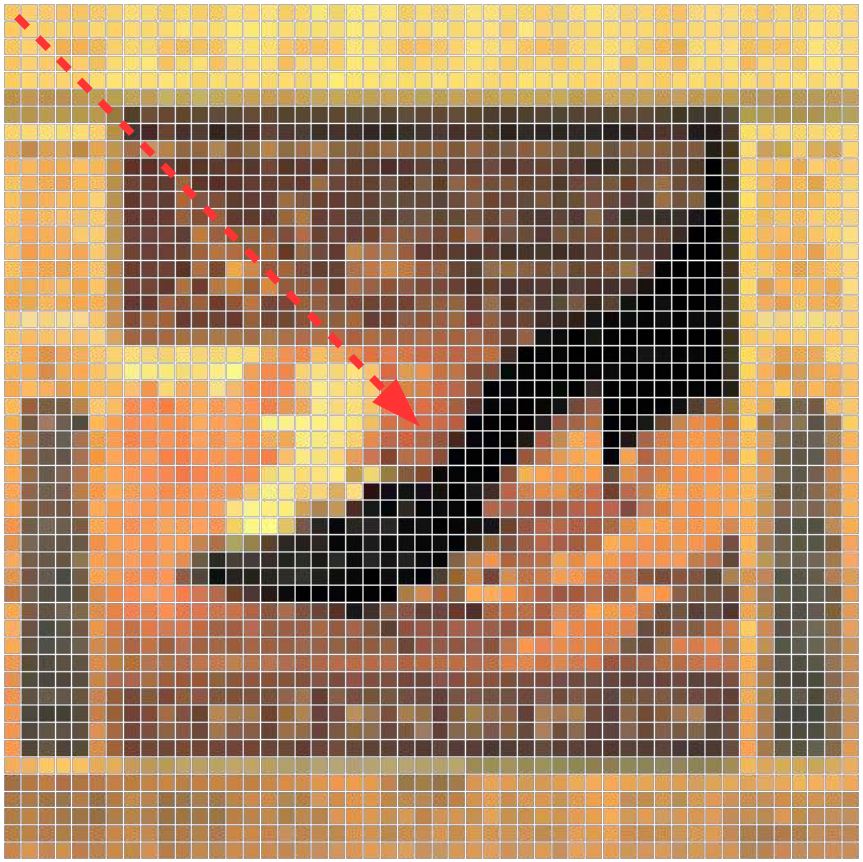


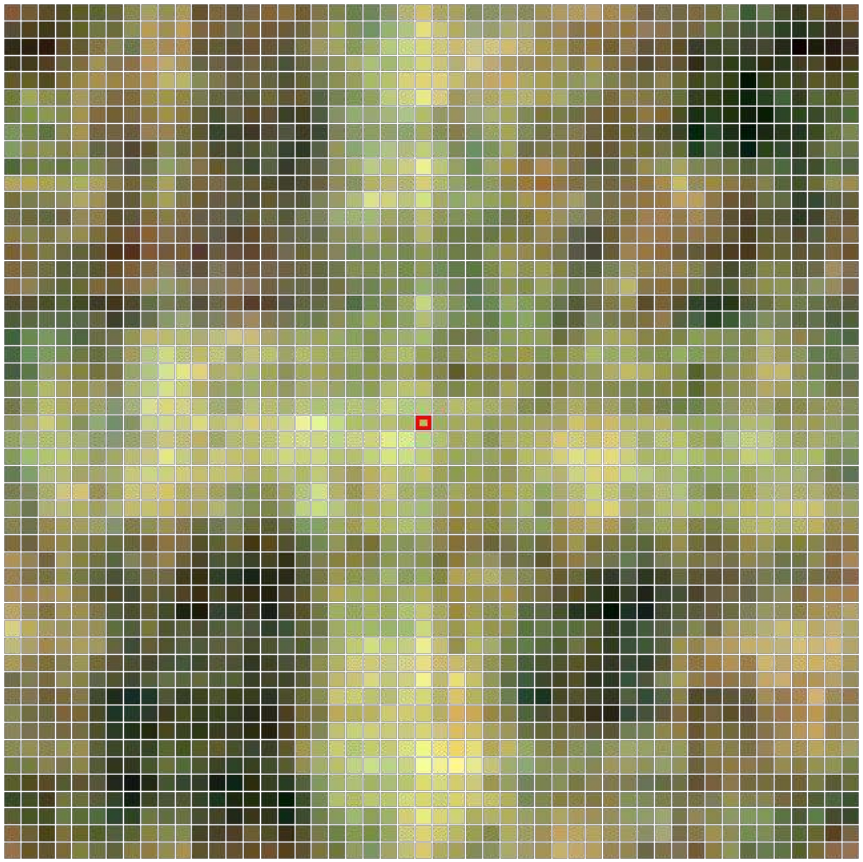
Our method — Translations identification — The case of images





Our method — Translations identification — The case of images





Injectivity

Translation operator « Black hole »

$$\forall v_1, v_2 \in \mathcal{V} : (\psi(v_1) = \psi(v_2) \neq \perp) \Rightarrow (v_1 = v_2)$$

Injectivity

Translation operator « Black hole »

$$\forall v_1, v_2 \in \mathcal{V} : (\psi(v_1) = \psi(v_2) \neq \perp) \Rightarrow (v_1 = v_2)$$

Restriction to the edges (EC)

$$\forall v \in \mathcal{V} : (\{v, \psi(v)\} \in \mathcal{E}) \vee (\psi(v) = \perp)$$

Injectivity

Translation operator « Black hole »

$$\forall v_1, v_2 \in \mathcal{V} : (\psi(v_1) = \psi(v_2) \neq \perp) \Rightarrow (v_1 = v_2)$$

Restriction to the edges (EC)

$$\forall v \in \mathcal{V} : (\{v, \psi(v)\} \in \mathcal{E}) \vee (\psi(v) = \perp)$$

Preservation of the neighborhood (SNP)

$$\forall v_1, v_2 \in \mathcal{V} : (\{v_1, v_2\} \in \mathcal{E} \Leftrightarrow \{\psi(v_1), \psi(v_2)\} \in \mathcal{E}) \vee (\psi(v_1) = \perp) \vee (\psi(v_2) = \perp)$$

Injectivity

Translation operator « Black hole »

$$\forall v_1, v_2 \in \mathcal{V} : (\psi(v_1) = \psi(v_2) \neq \perp) \Rightarrow (v_1 = v_2)$$

Restriction to the edges (EC)

$$\forall v \in \mathcal{V} : (\{v, \psi(v)\} \in \mathcal{E}) \vee (\psi(v) = \perp)$$

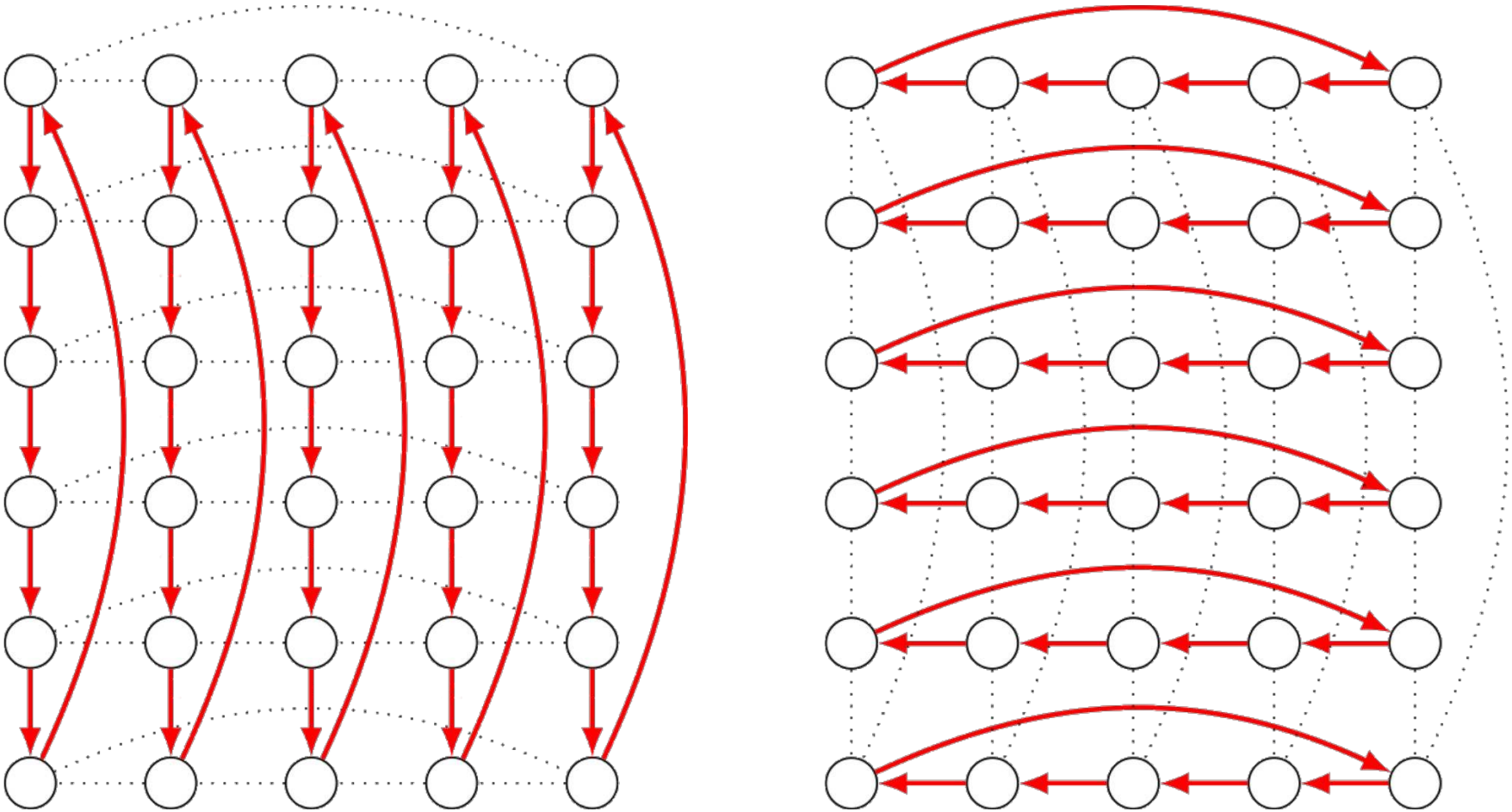
Preservation of the neighborhood (SNP)

$$\forall v_1, v_2 \in \mathcal{V} : (\{v_1, v_2\} \in \mathcal{E} \Leftrightarrow \{\psi(v_1), \psi(v_2)\} \in \mathcal{E}) \vee (\psi(v_1) = \perp) \vee (\psi(v_2) = \perp)$$

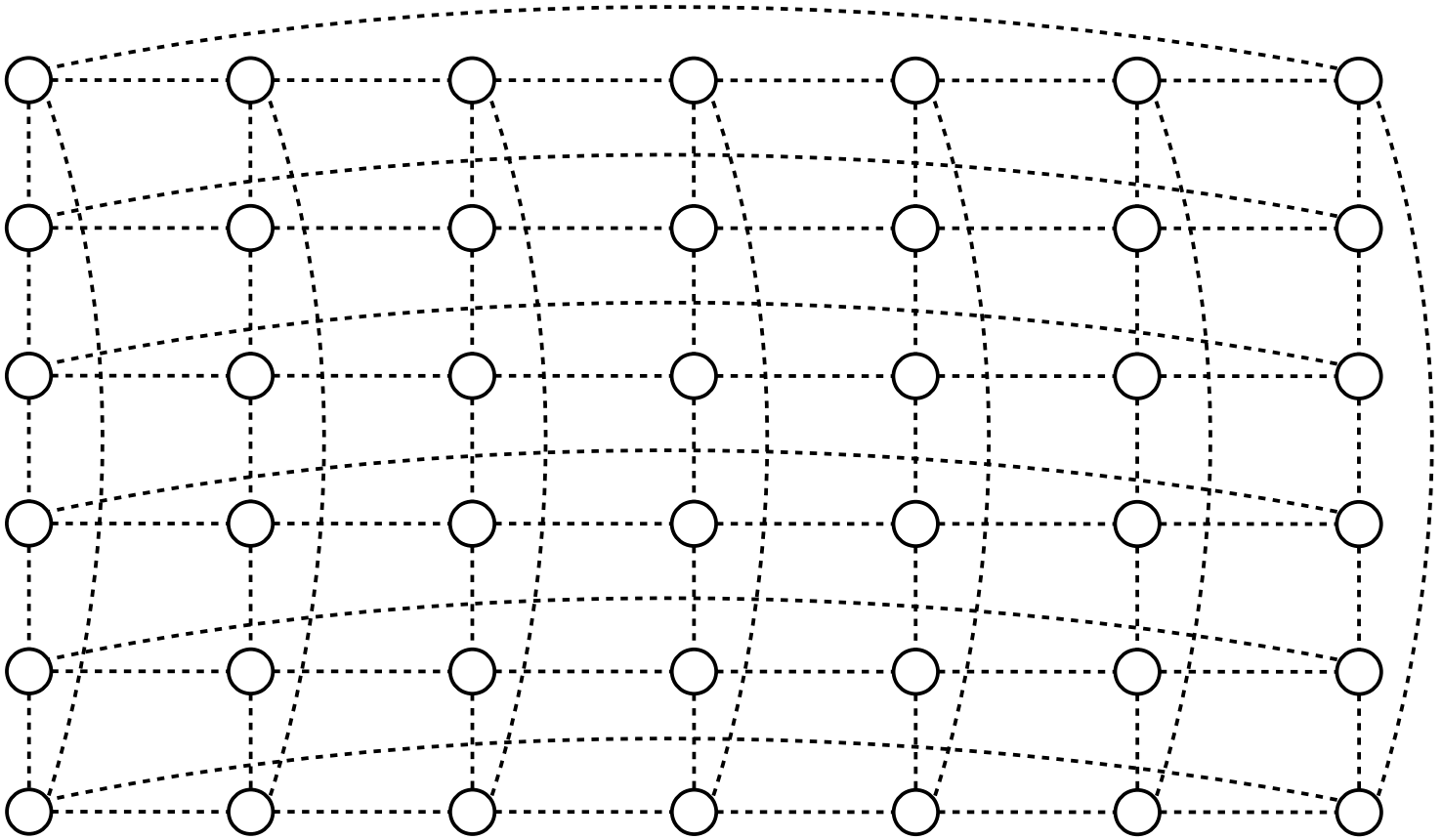
!

Preserves locality/structure of a signal
Perfect for convolutional kernels

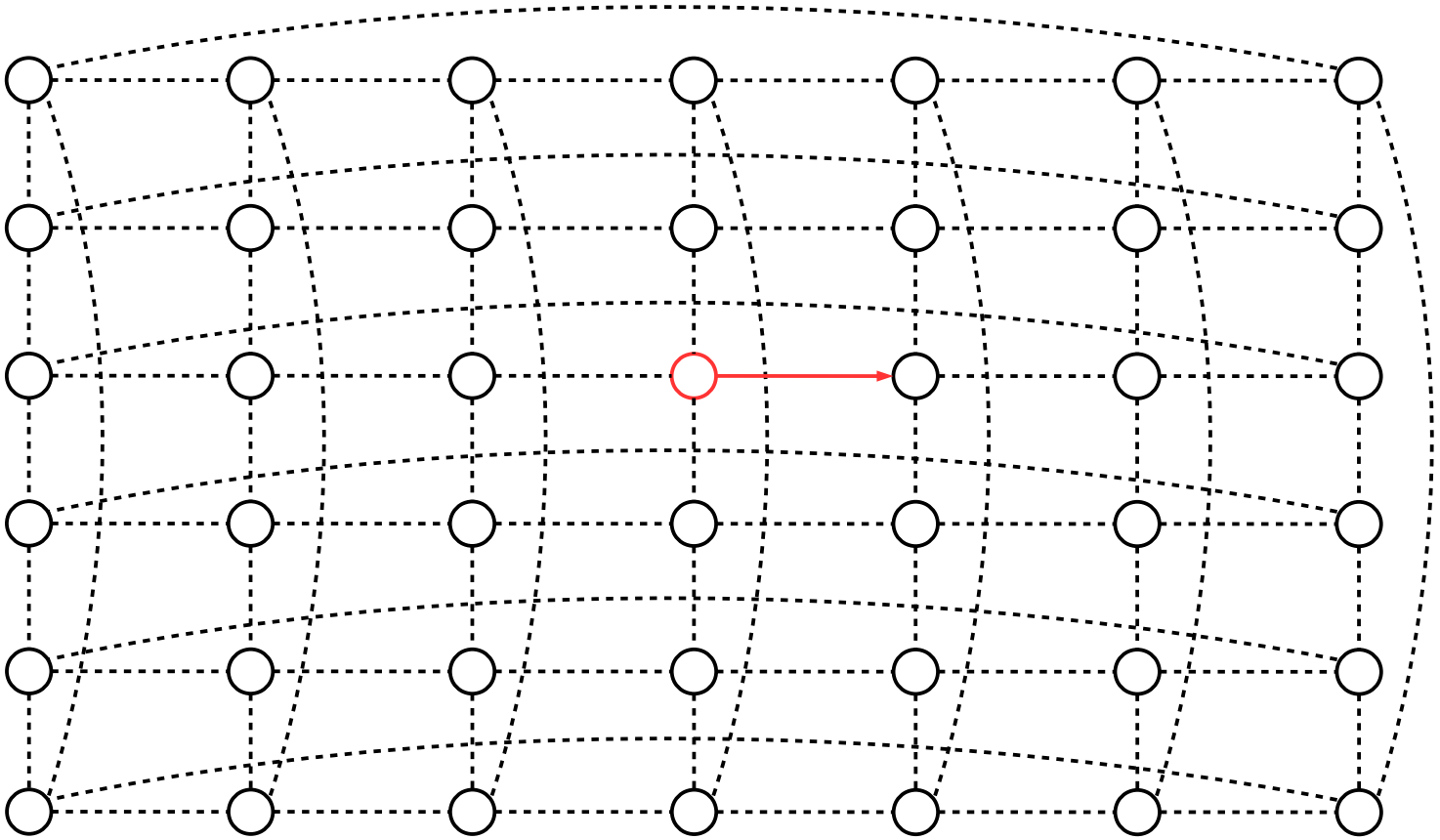
Our method — Translations identification — On the torus graph



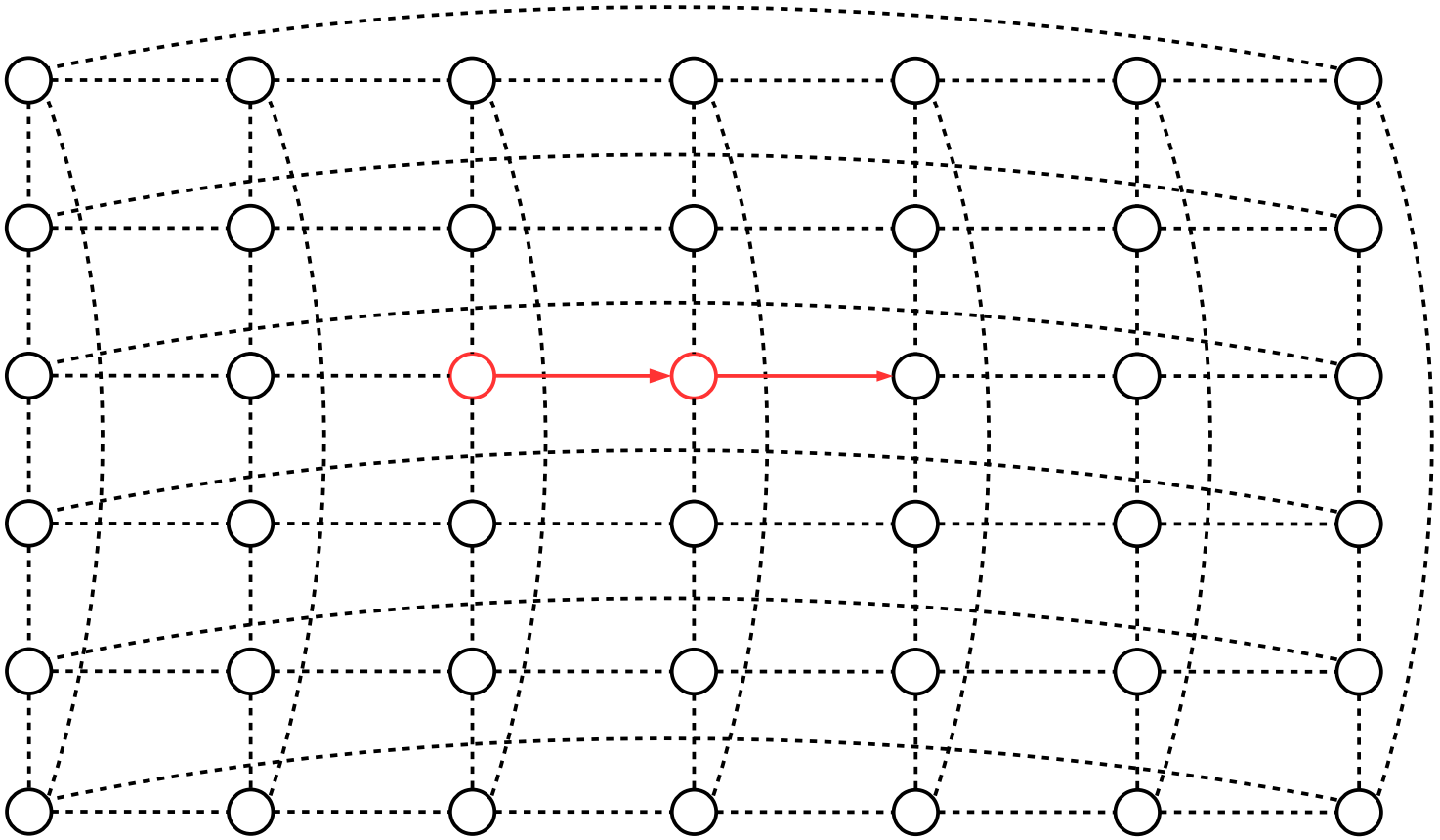
Our method — Translations identification — Sketch of the proof



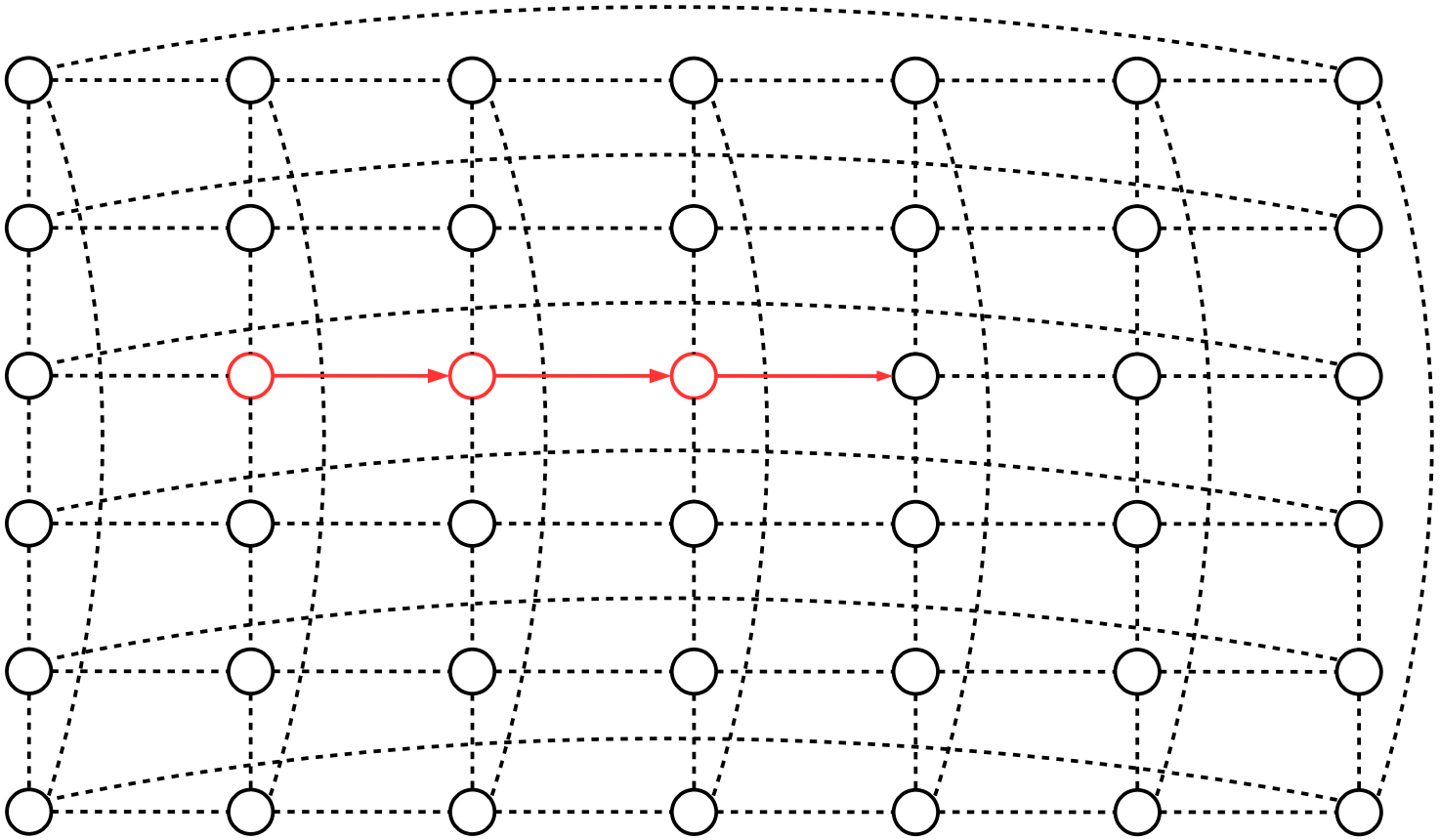
Our method — Translations identification — Sketch of the proof



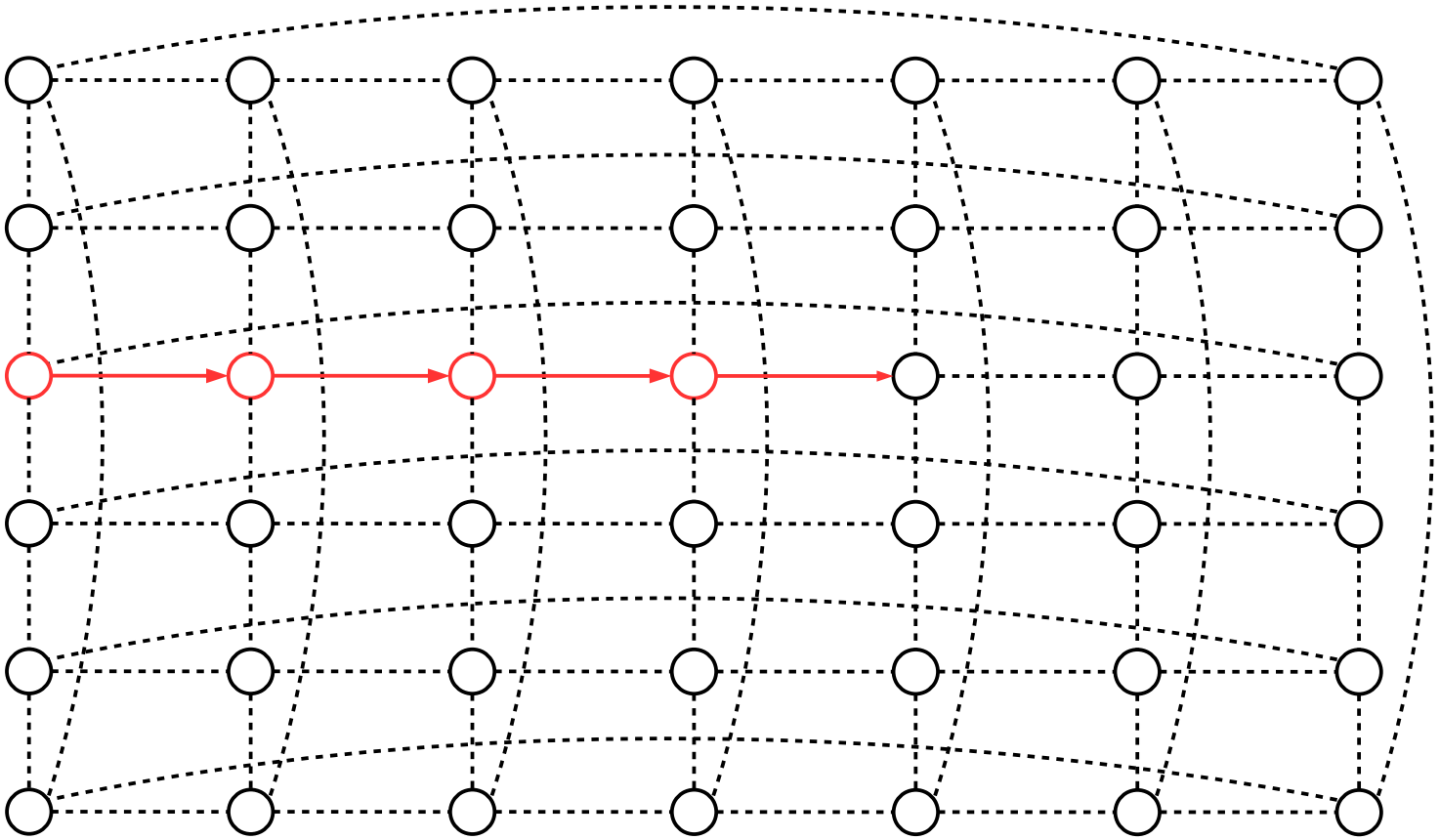
Our method — Translations identification — Sketch of the proof



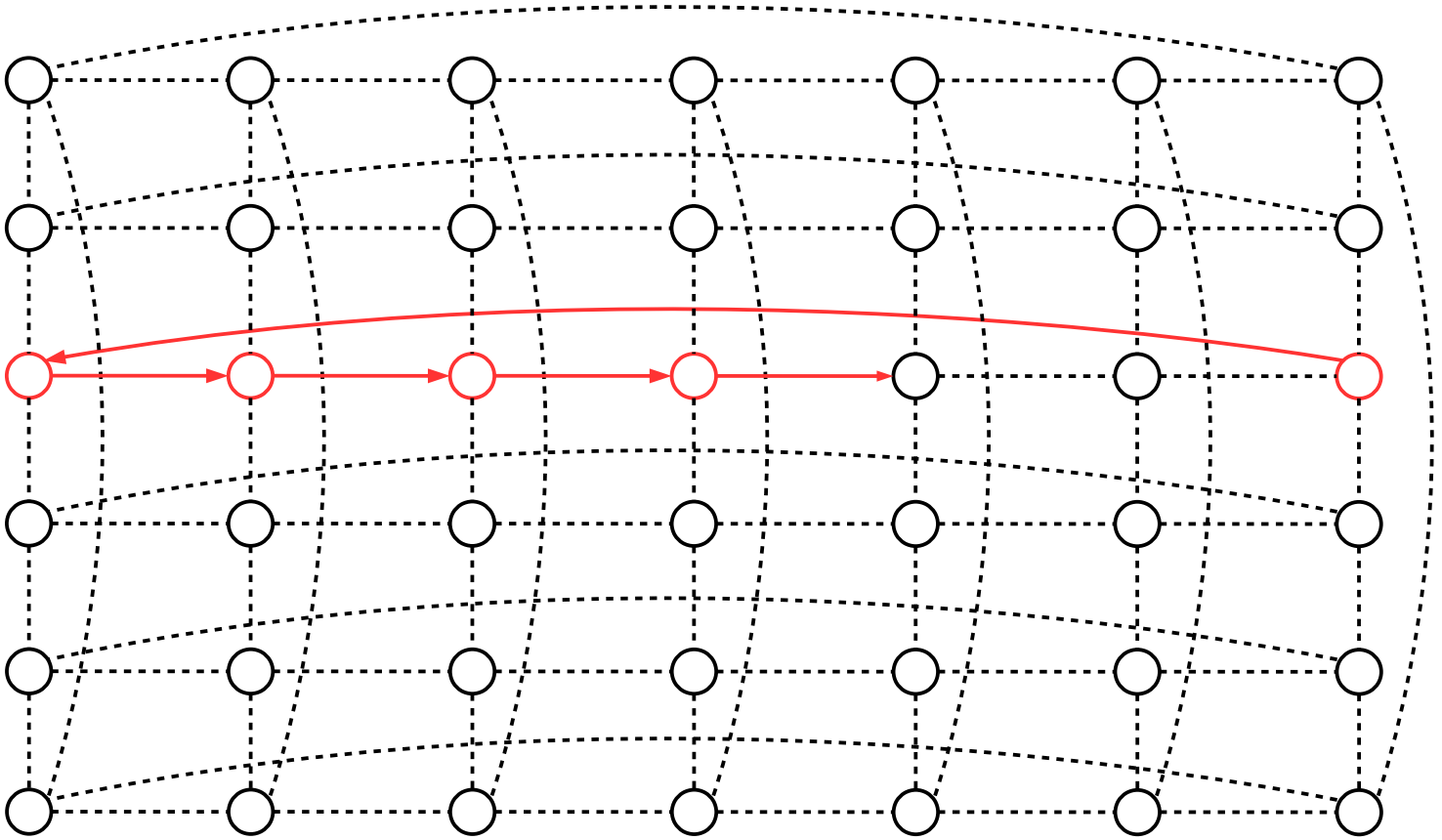
Our method — Translations identification — Sketch of the proof



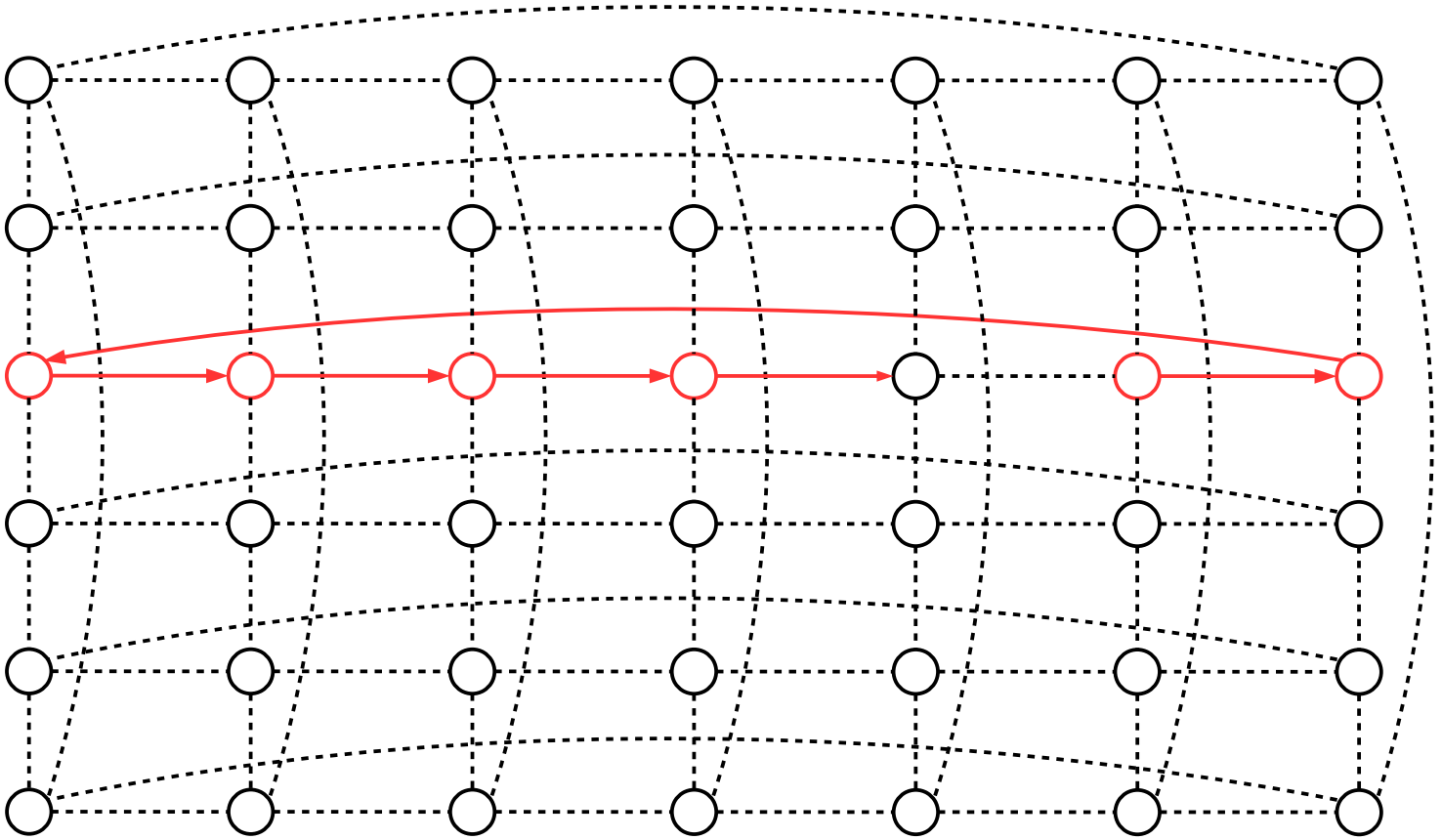
Our method — Translations identification — Sketch of the proof



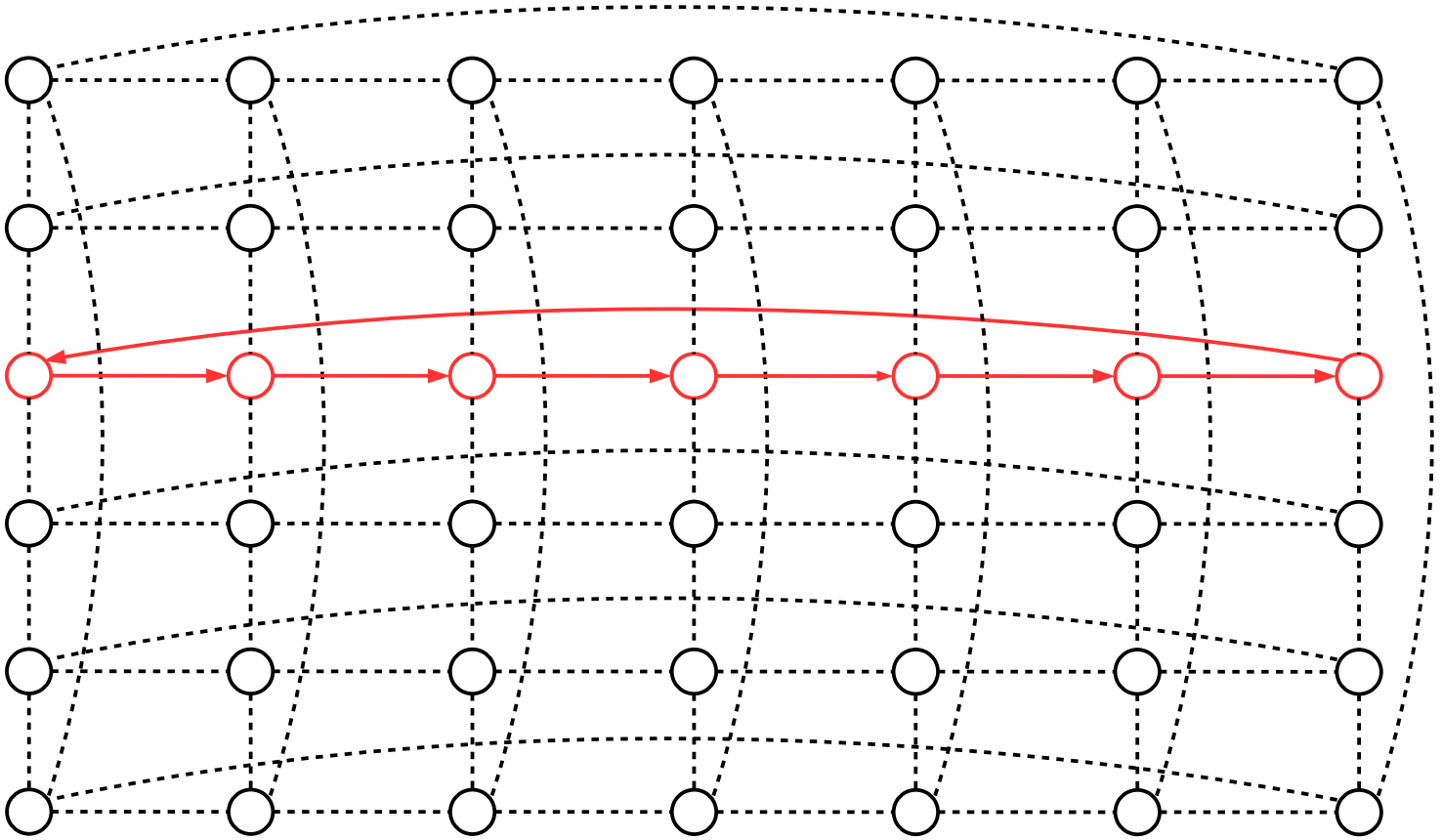
Our method — Translations identification — Sketch of the proof



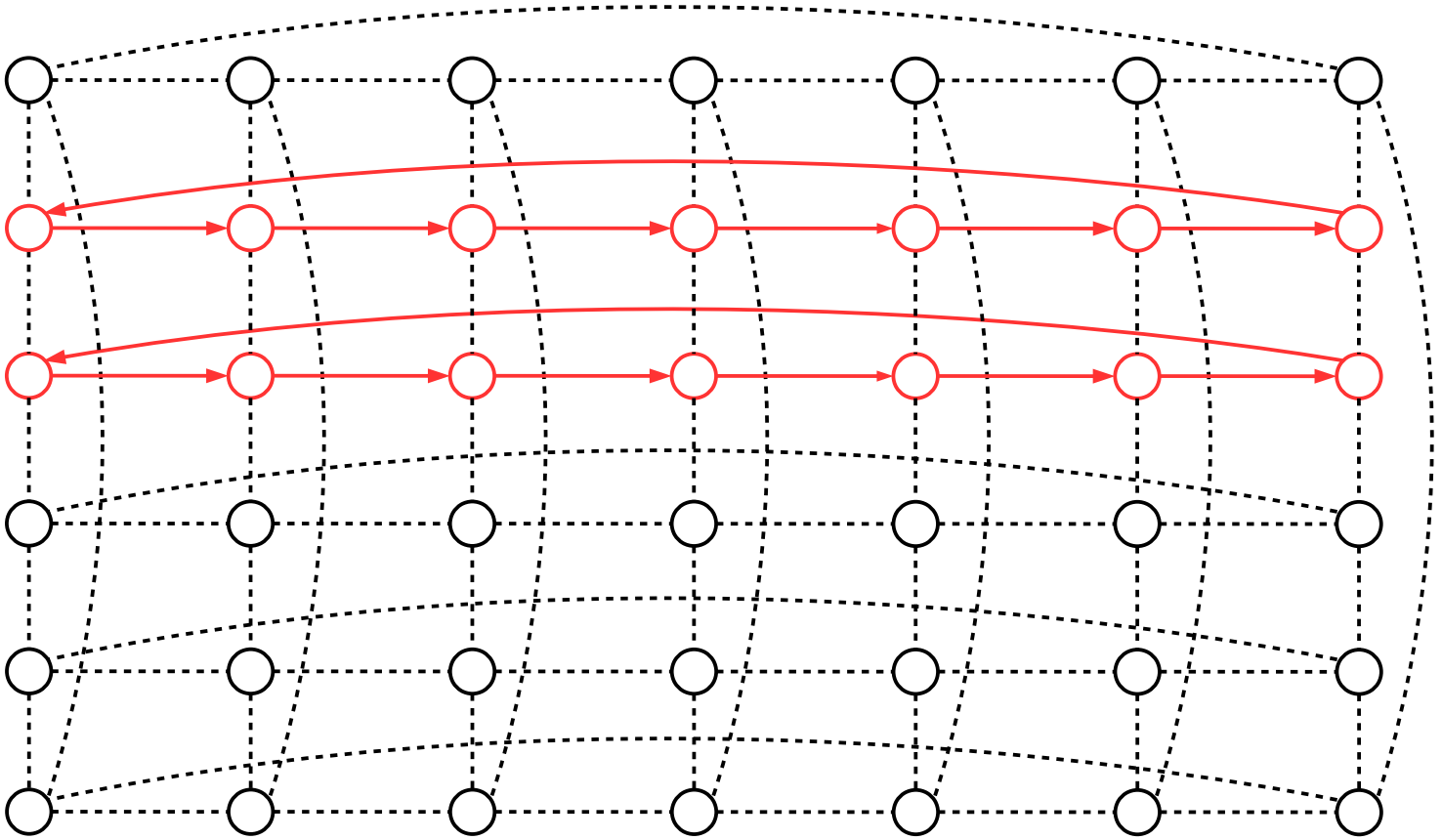
Our method — Translations identification — Sketch of the proof



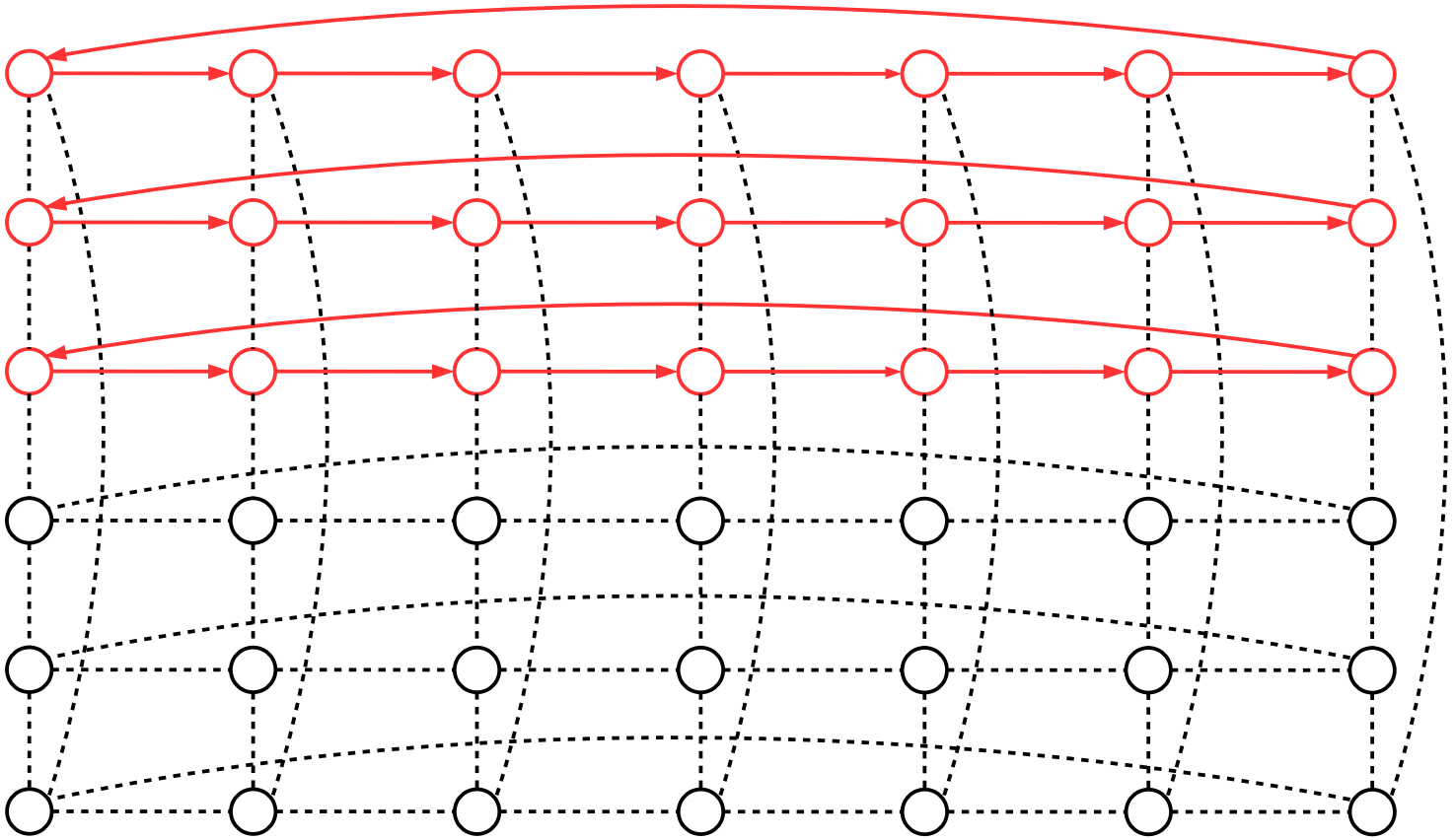
Our method — Translations identification — Sketch of the proof



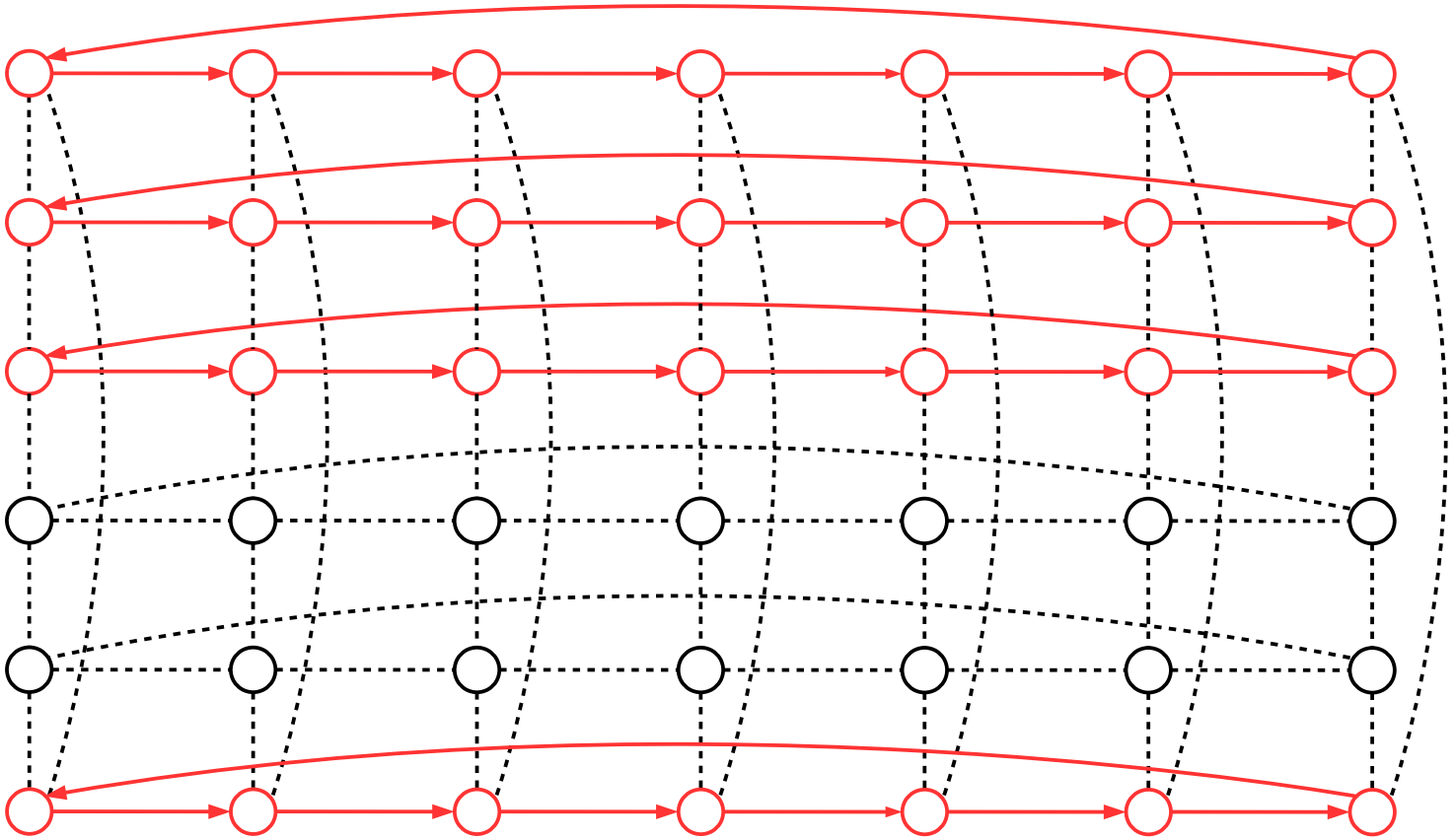
Our method — Translations identification — Sketch of the proof



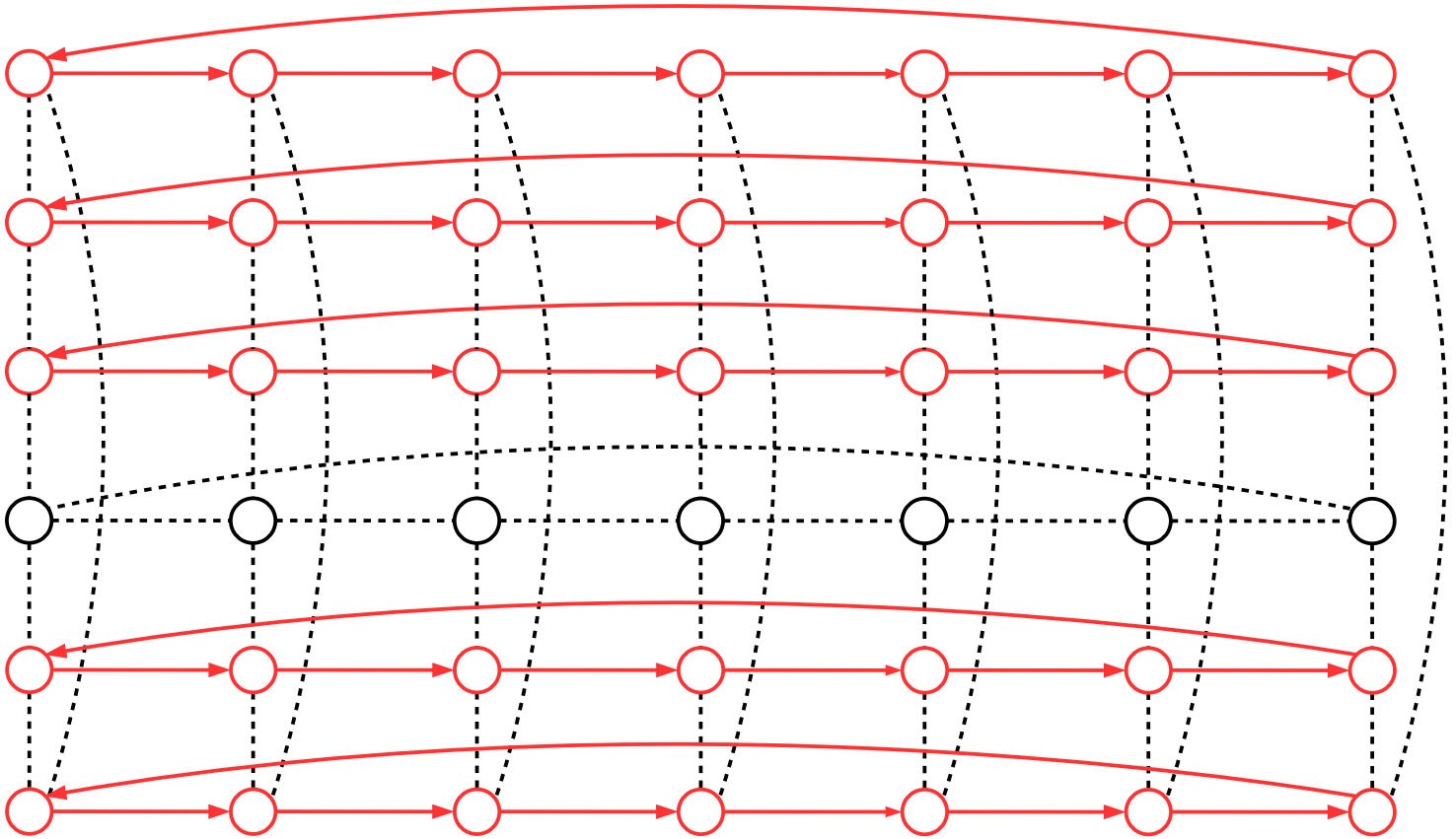
Our method — Translations identification — Sketch of the proof



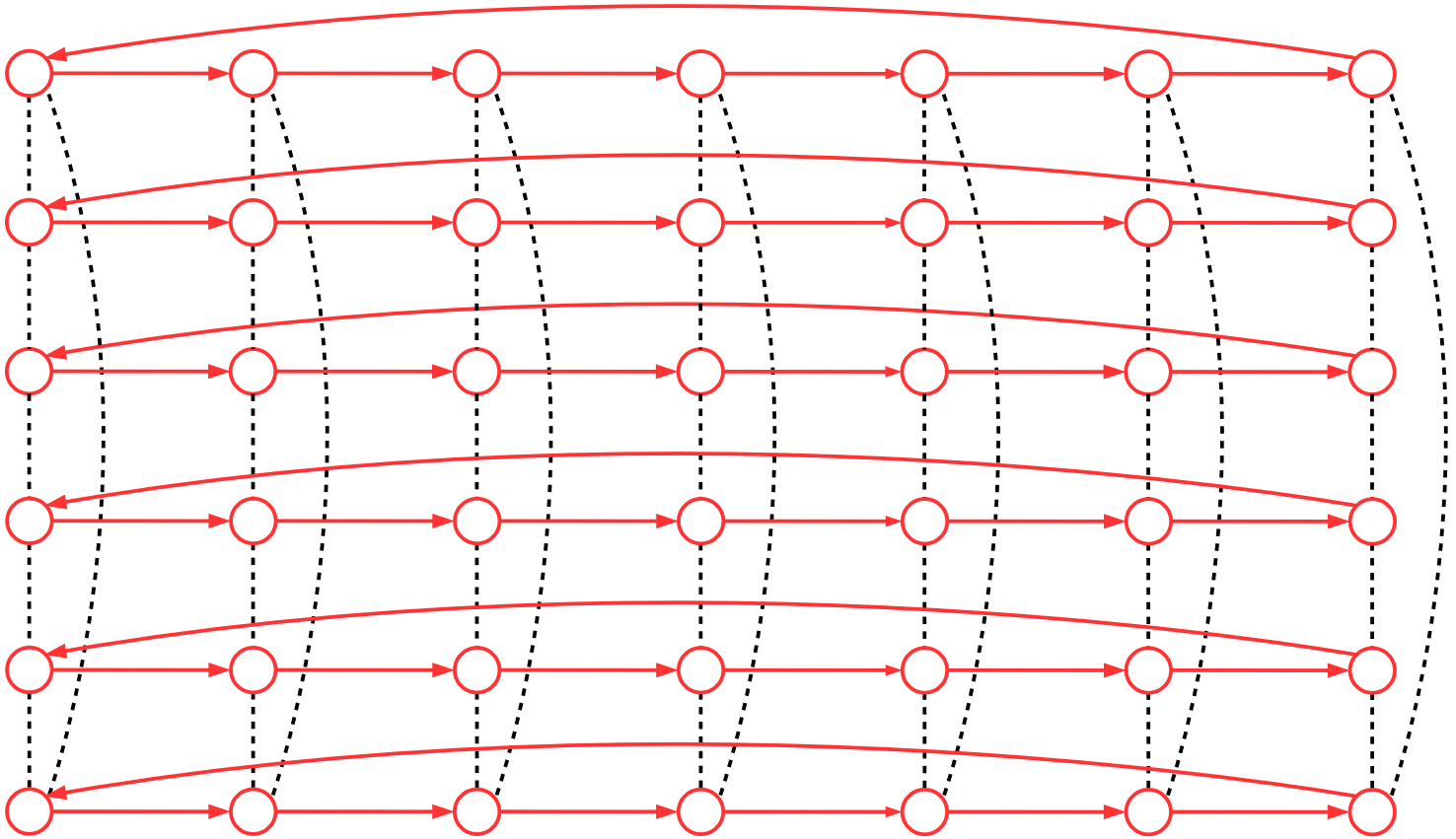
Our method — Translations identification — Sketch of the proof



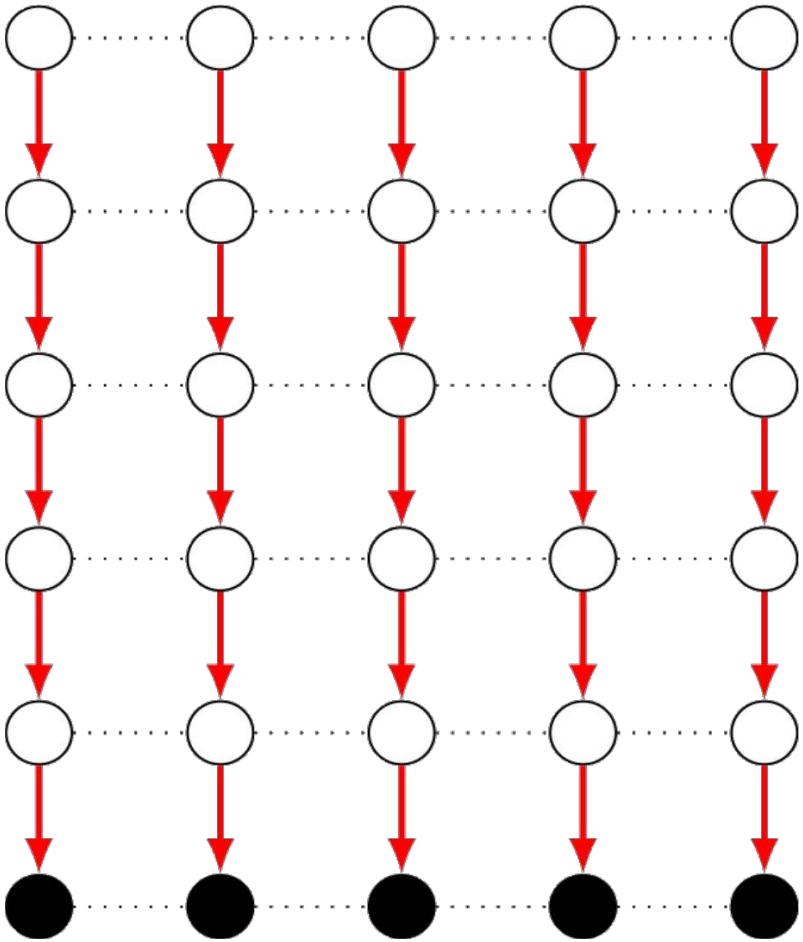
Our method — Translations identification — Sketch of the proof

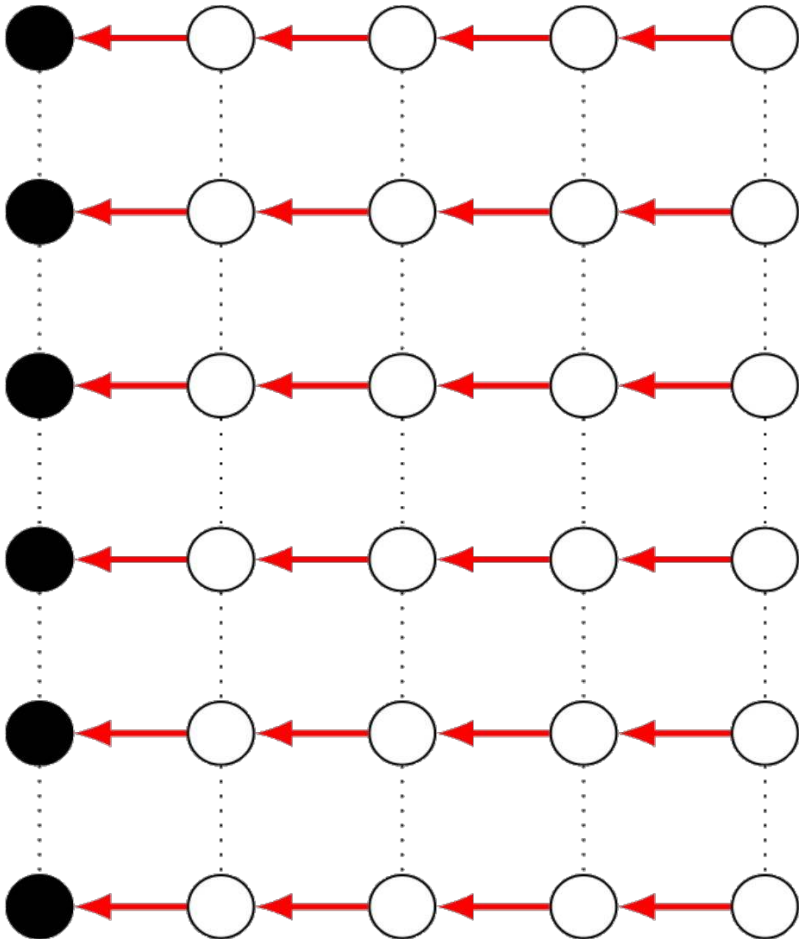


Our method — Translations identification — Sketch of the proof

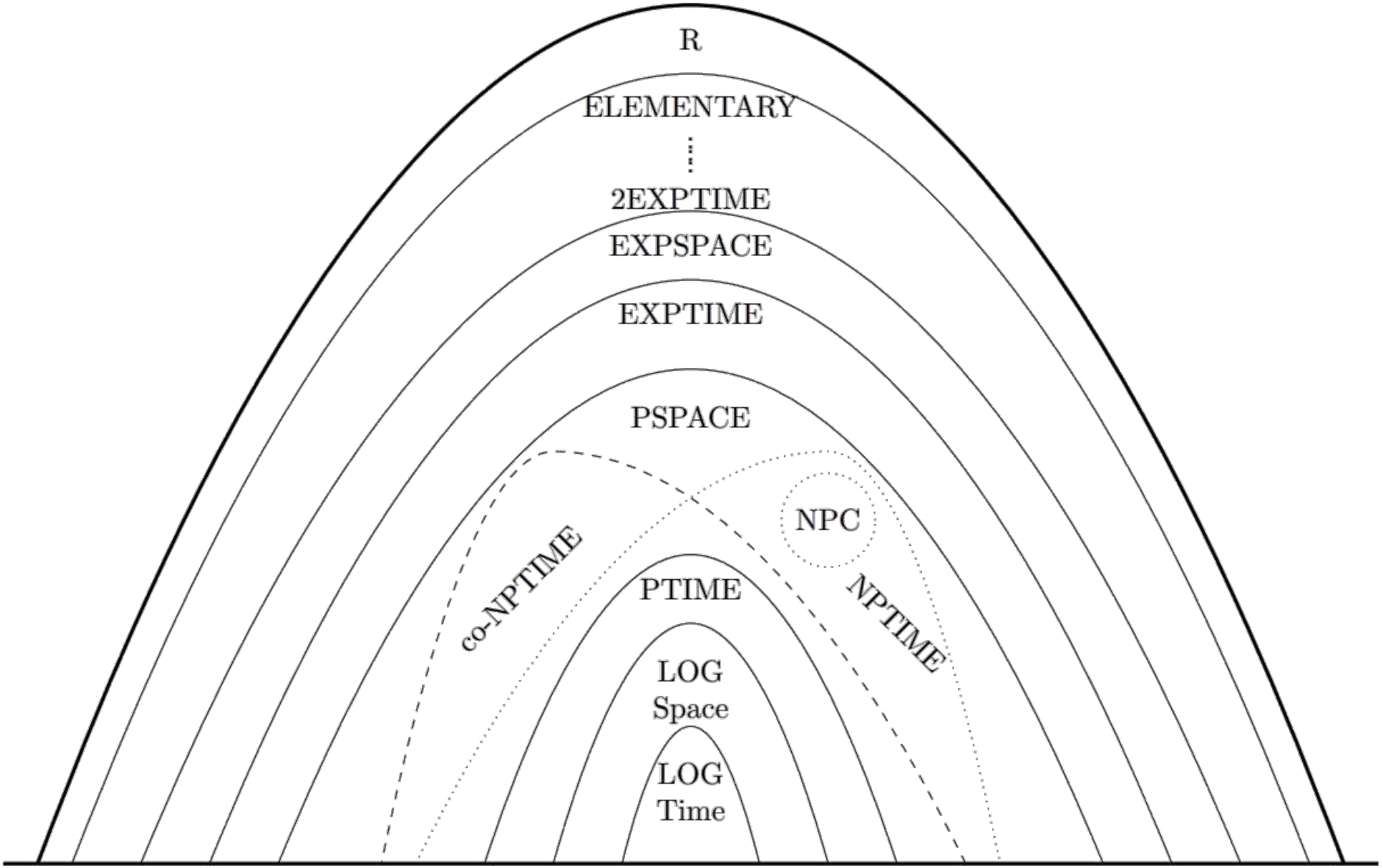


Our method — Translations identification — On the grid graph

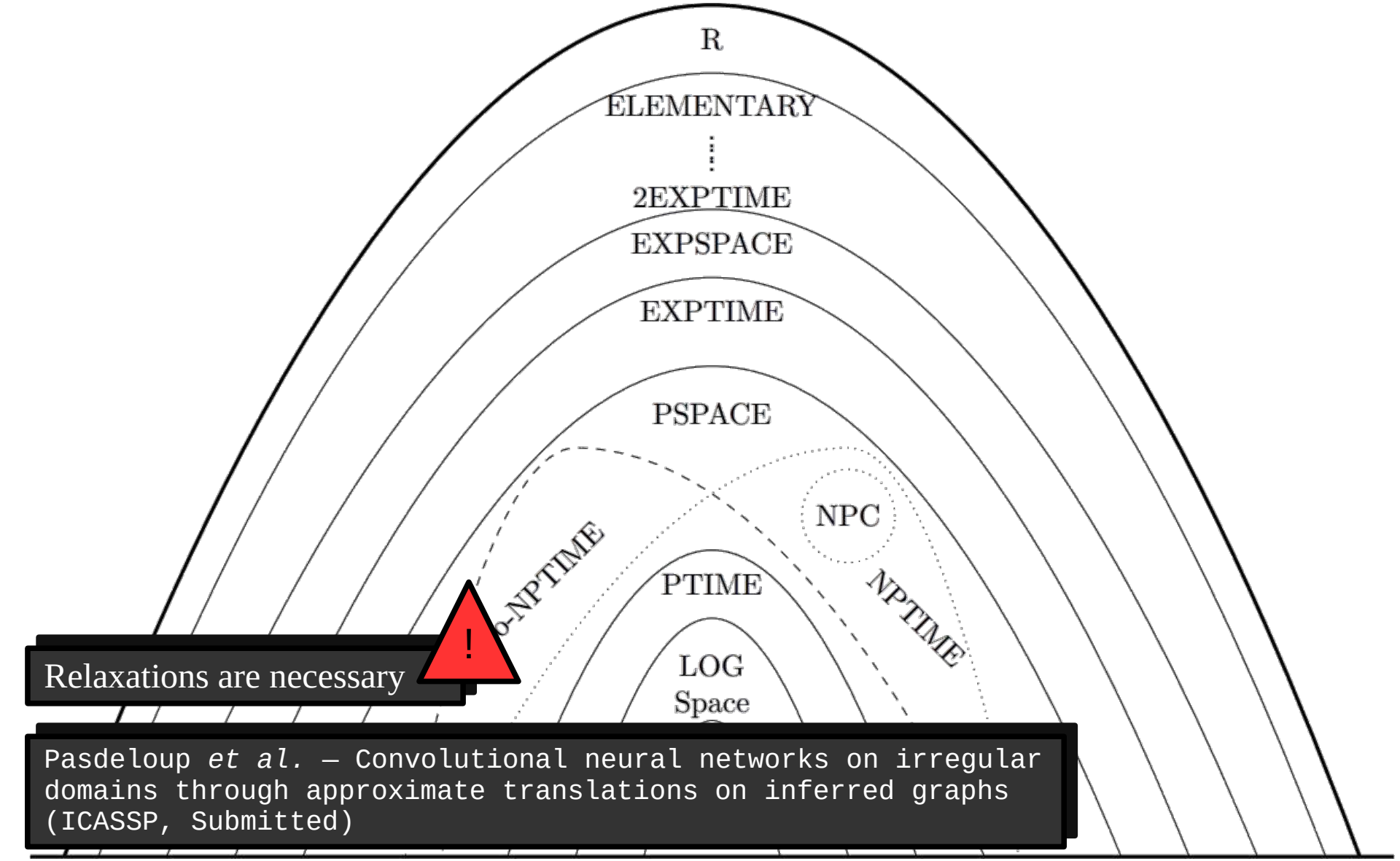




Our method — Translations identification — A difficult problem



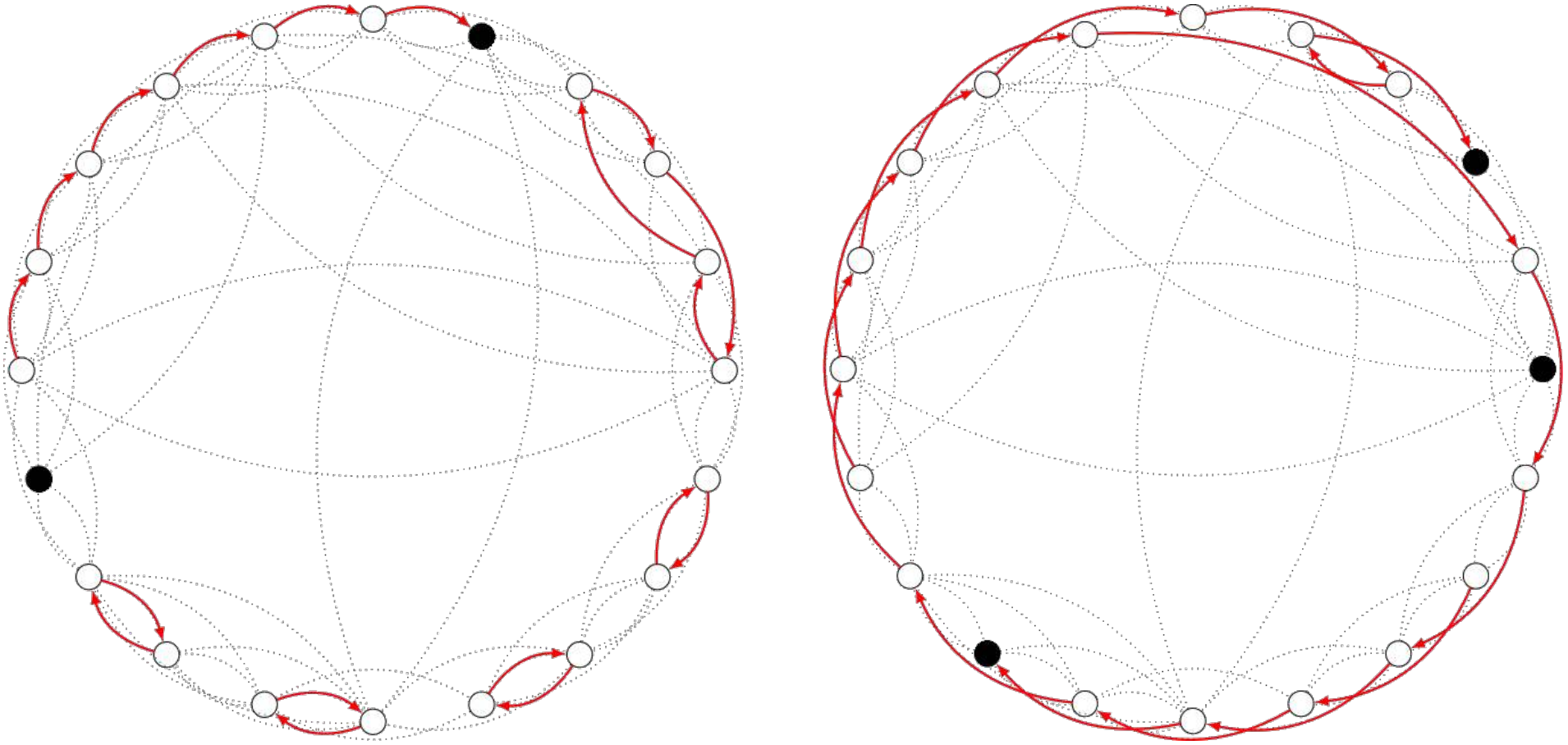
Our method — Translations identification — A difficult problem



Relaxations are necessary

Pasdeloup *et al.* — Convolutional neural networks on irregular domains through approximate translations on inferred graphs (ICASSP, Submitted)

Our method — Translations identification — Watts-Strogatz model



Outline

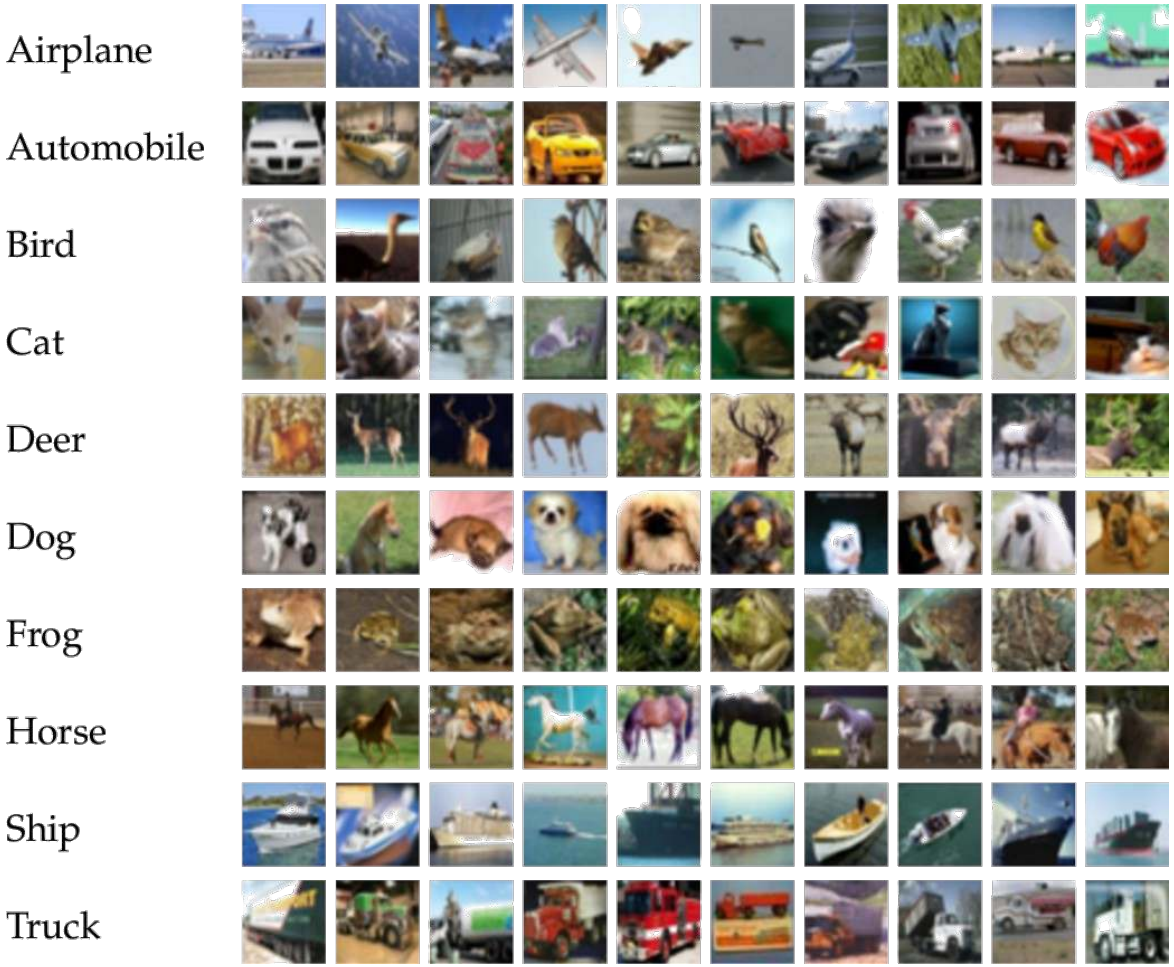
Existing methods & required notions

The method we propose

— Global idea of the method —
— Graph inference from data —
— Identification of translations —

Results

Results — The CIFAR-10 dataset



Dense
72.7 %

Convolutional
94.81 %

Data augmentation
97.28 %

Cheng *et al.* — Deep Haar scattering networks (2016)

Results — The graph inference methods used

Covariance + max K per row + symmetrization ..... $\mathcal{G}_{\tilde{\Sigma}}$

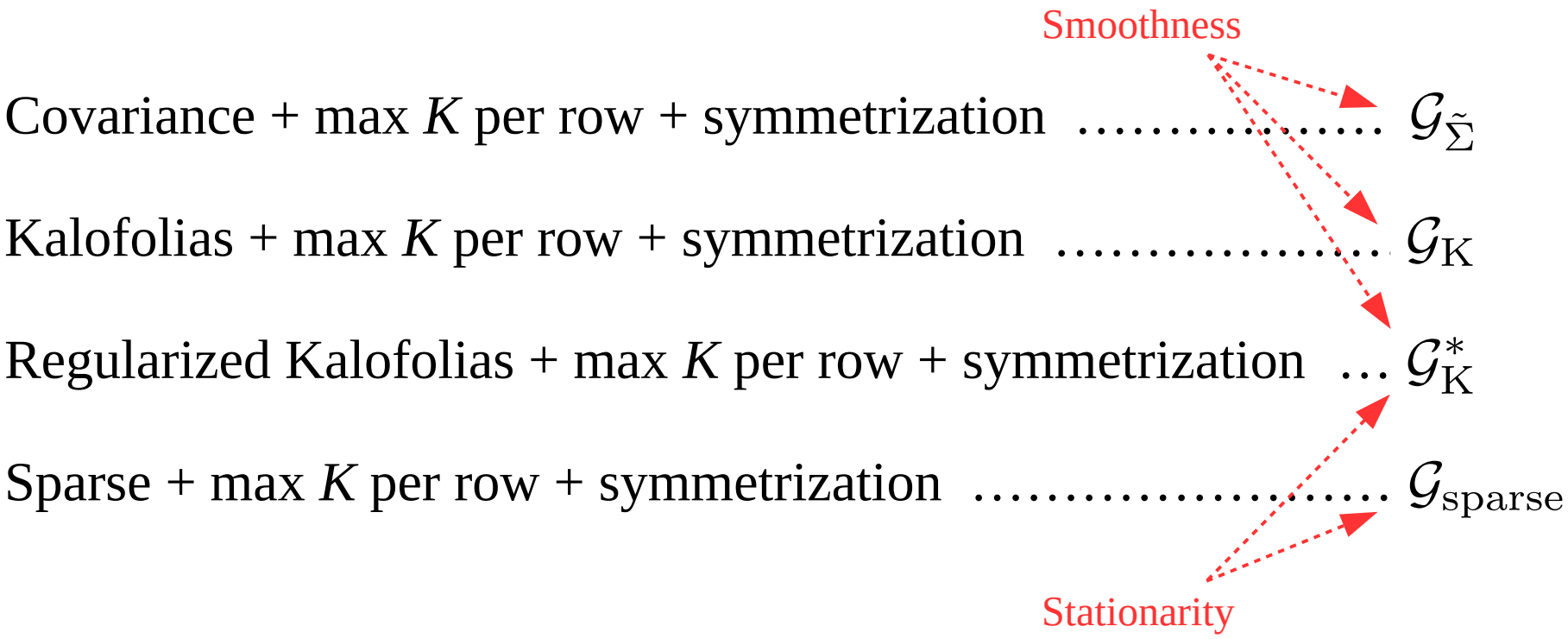
Kalofolias + max K per row + symmetrization ..... $\mathcal{G}_{\mathrm{K}}$

Regularized Kalofolias + max K per row + symmetrization ... $\mathcal{G}_{\mathrm{K}}^{*}$

Sparse + max K per row + symmetrization ..... $\mathcal{G}_{\mathrm{sparse}}$

Baseline (3x3 sliding kernel) ..... $\mathcal{G}_{g}$

Results — The graph inference methods used

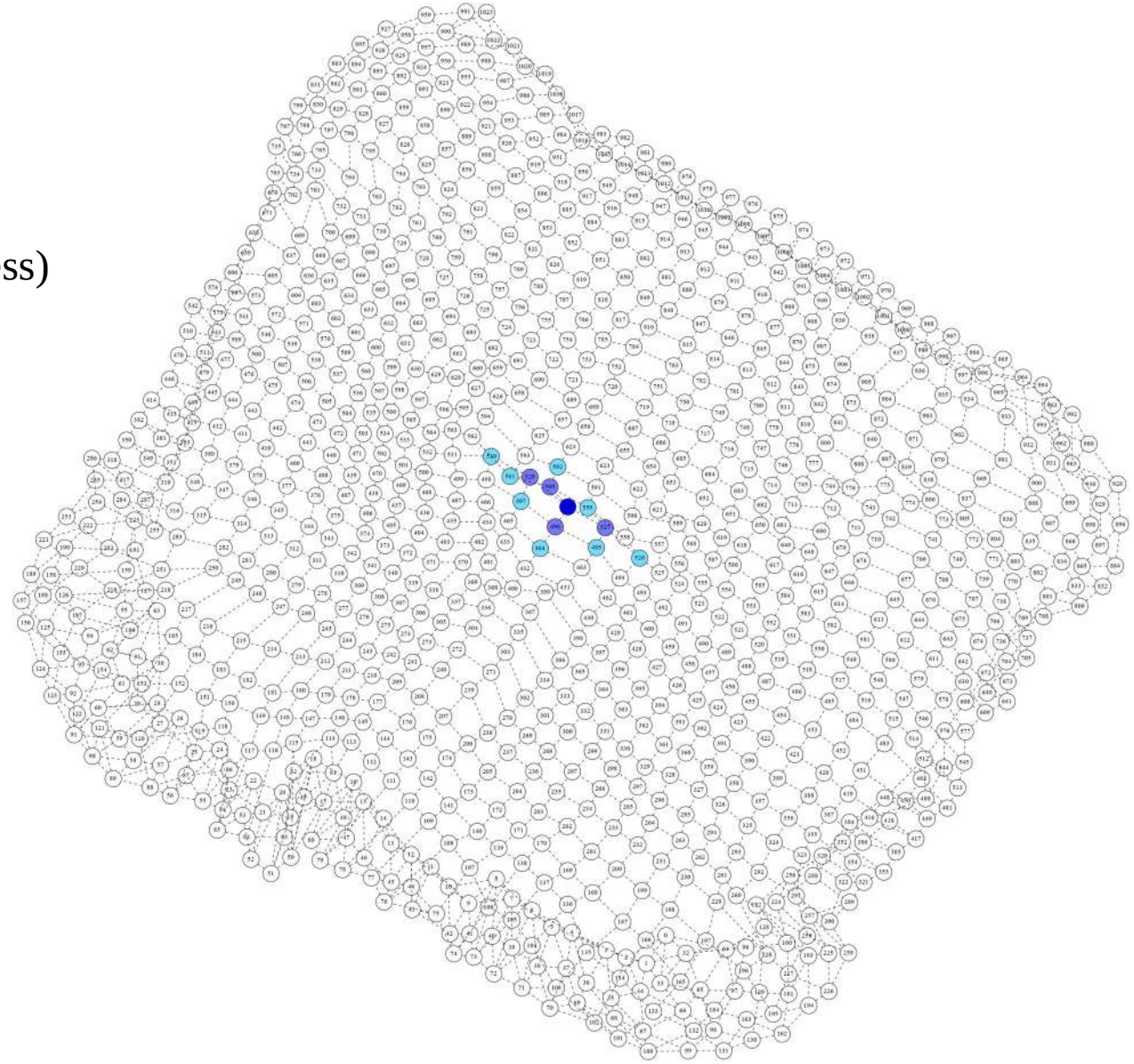


Baseline (3x3 sliding kernel) ..... $\mathcal{G}_g$

Results — Some of the inferred graphs

$\mathcal{G}_{\tilde{\Sigma}}$

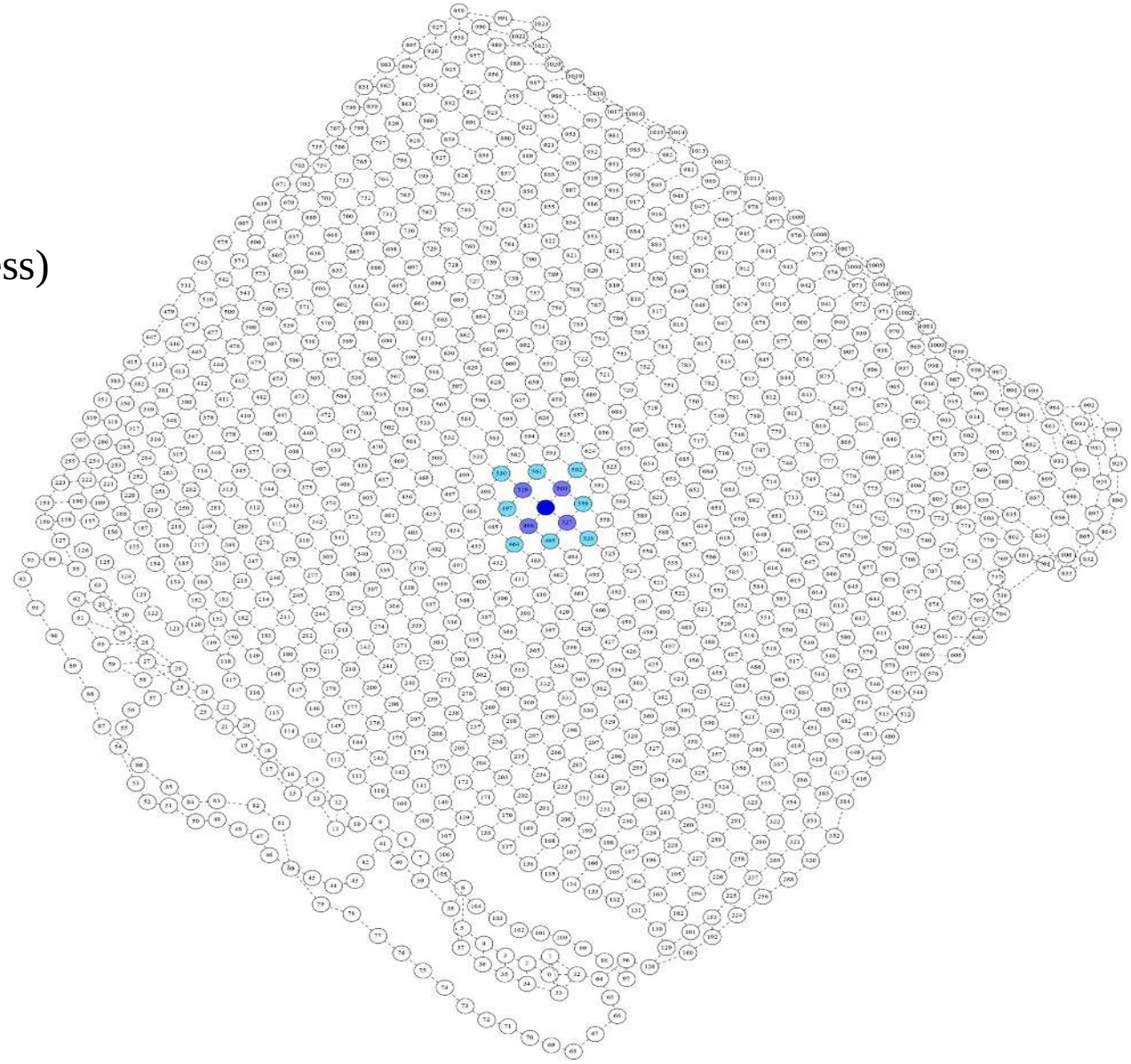
(smoothness)



Results — Some of the inferred graphs

$\mathcal{G}_{\mathrm{K}}$

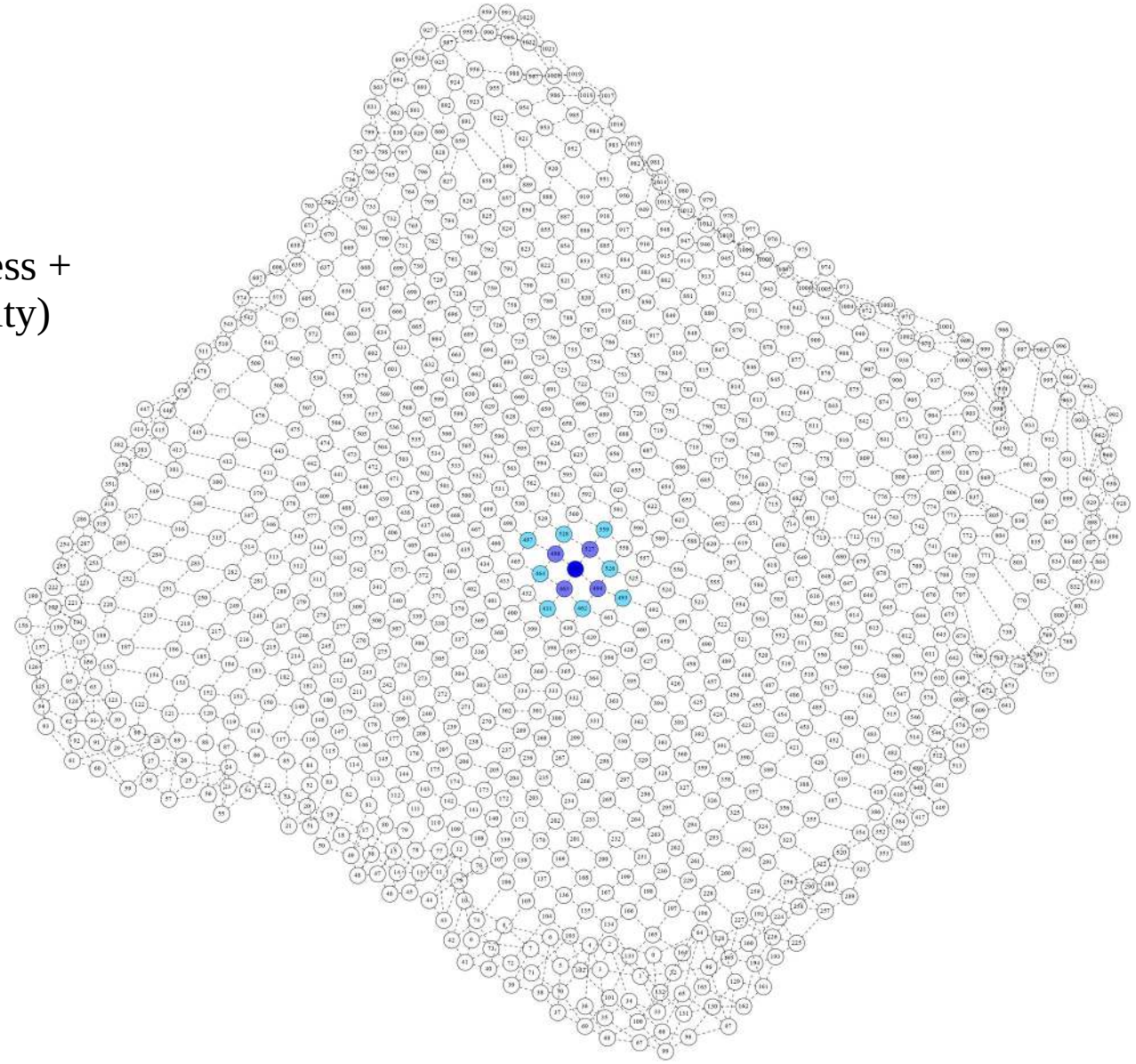
(smoothness)



Results — Some of the inferred graphs

$\mathcal{G}_{\mathrm{K}}^{*}$

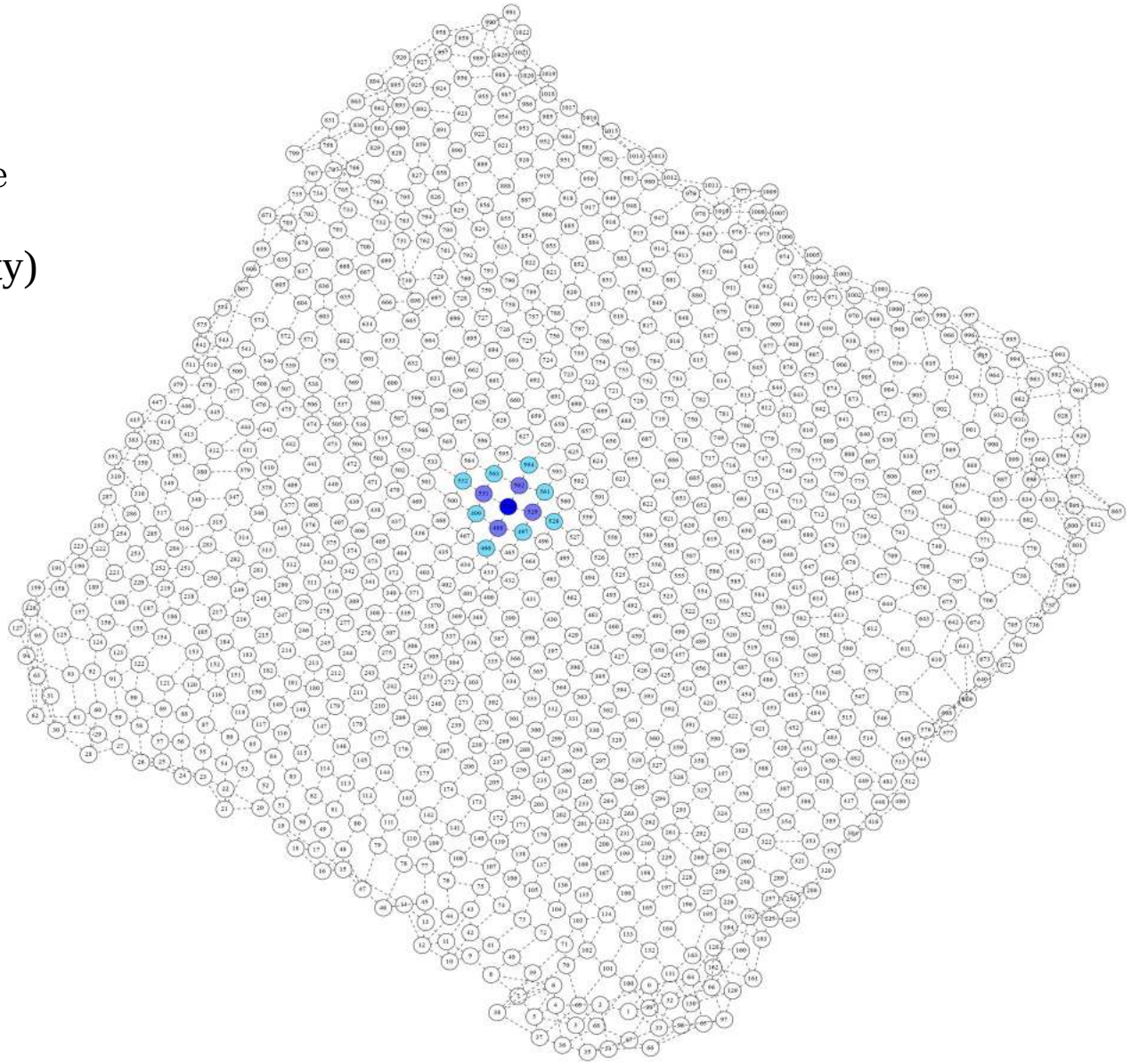
(smoothness + stationarity)



Results — Some of the inferred graphs

$\mathcal{G}_{\text{sparse}}$

(stationarity)



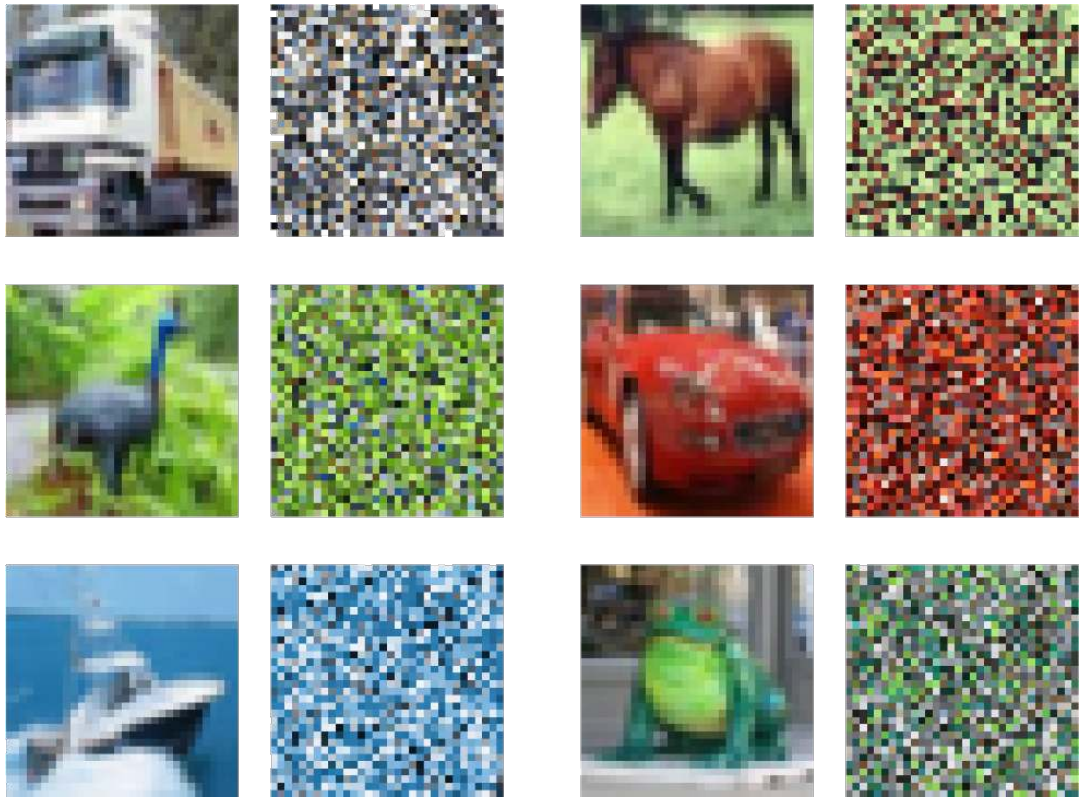
Results — Obtained results

$\mathcal{G}_g$	$\mathcal{G}_{\widetilde{\Sigma}}$	$\mathcal{G}_{\mathrm{K}}$	$\mathcal{G}_{\mathrm{K}}^{*}$	$\mathcal{G}_{\mathrm{sparse}}$
85.98%	85.03%	85.33%	85.81%	77.54%

Best without using knowledge 72.7 %

Results — On the scrambled version of CIFAR-10

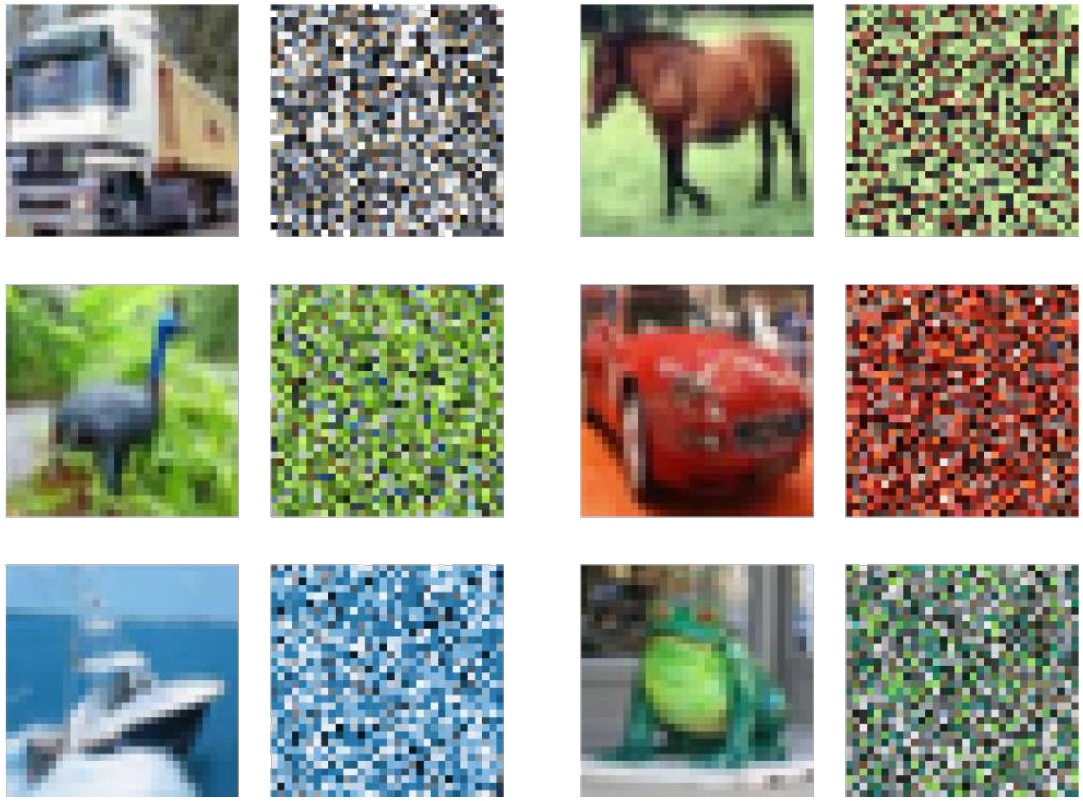
$\mathcal{G}_g$	$\mathcal{G}_{\widetilde{\mathbf{\Sigma}}}$	$\mathcal{G}_{\mathrm{K}}$	$\mathcal{G}_{\mathrm{K}}^*$	$\mathcal{G}_{\mathrm{sparse}}$
85.98%	85.03%	85.33%	85.81%	77.54%



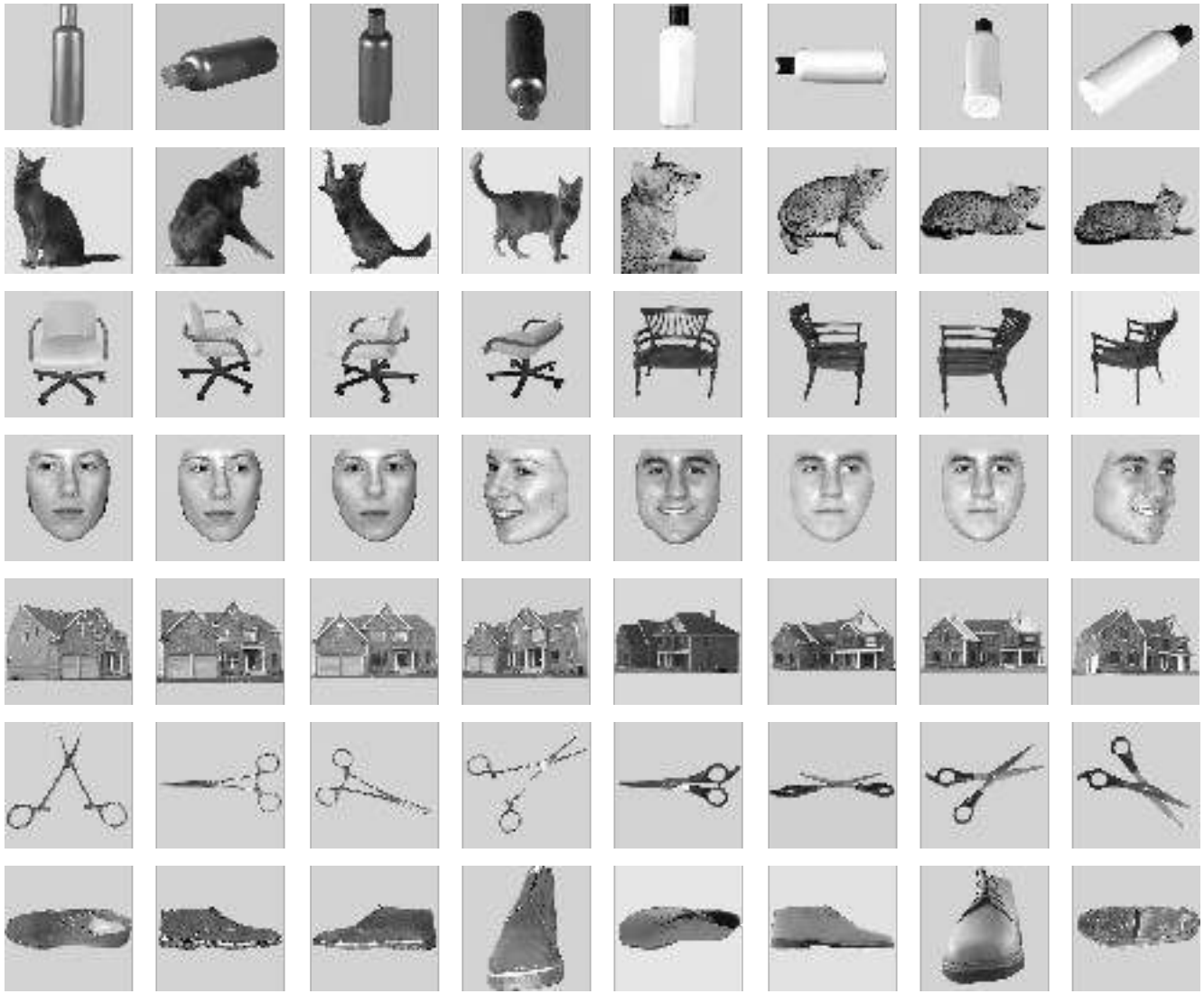
Results — On the scrambled version of CIFAR-10

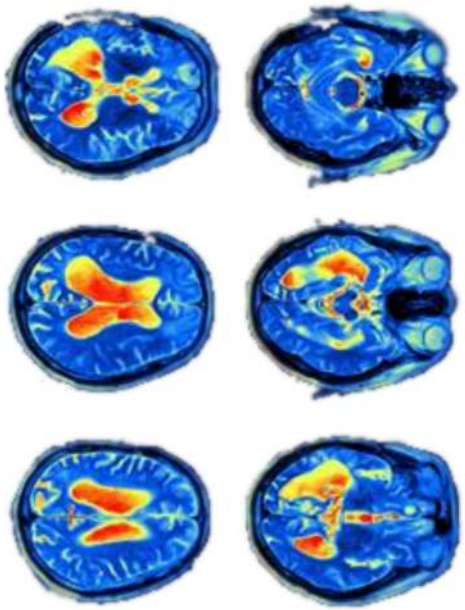
$\mathcal{G}_g$	$\mathcal{G}_{\widetilde{\Sigma}}$	$\mathcal{G}_{\mathrm{K}}$	$\mathcal{G}_{\mathrm{K}}^*$	$\mathcal{G}_{\mathrm{sparse}}$
85.98%	85.03%	85.33%	85.81%	77.54%

56.74 %



Results — The Haxby dataset

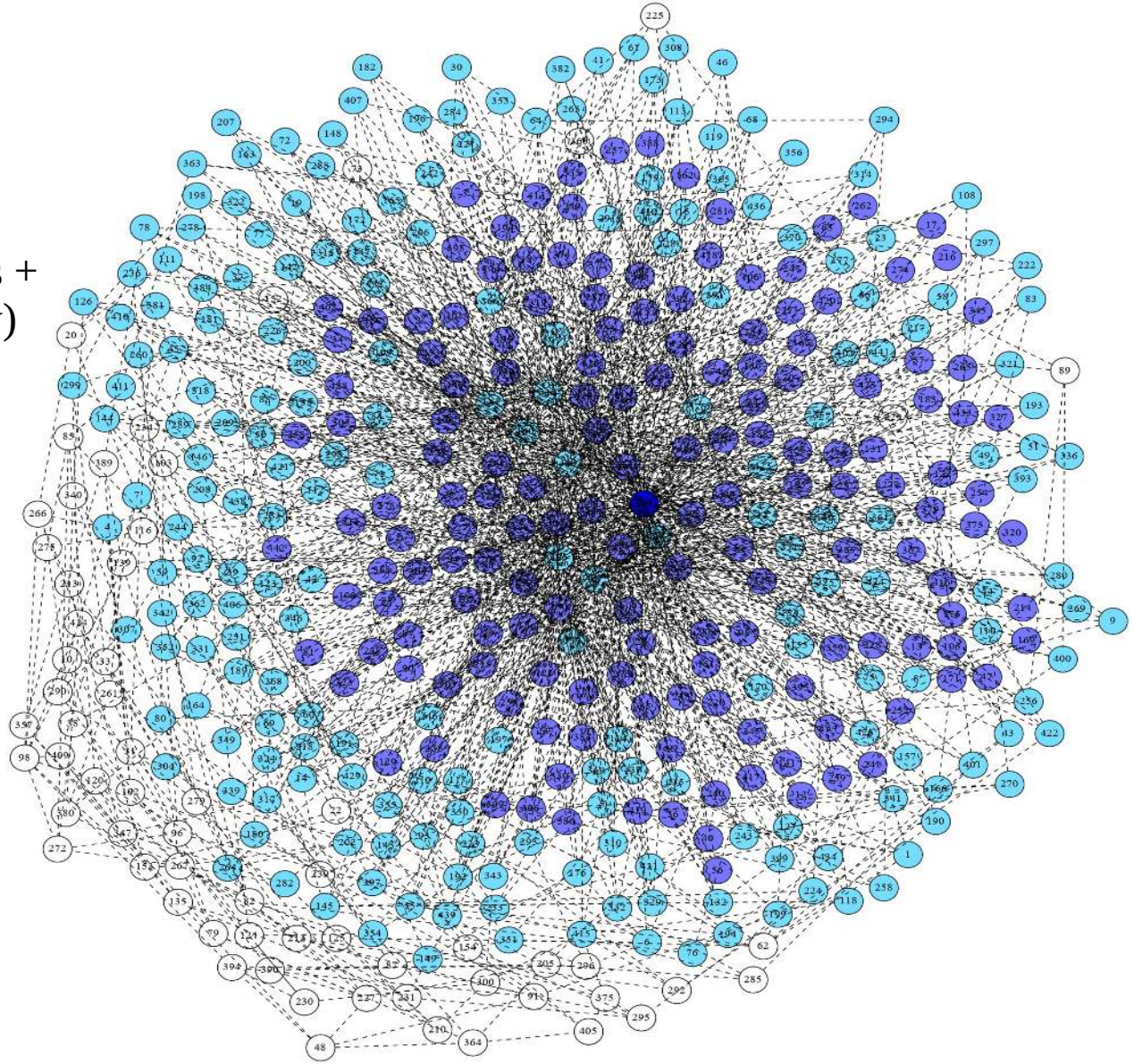




Results — The Haxby dataset

$\mathcal{G}_{\mathrm{K}}^{*}$

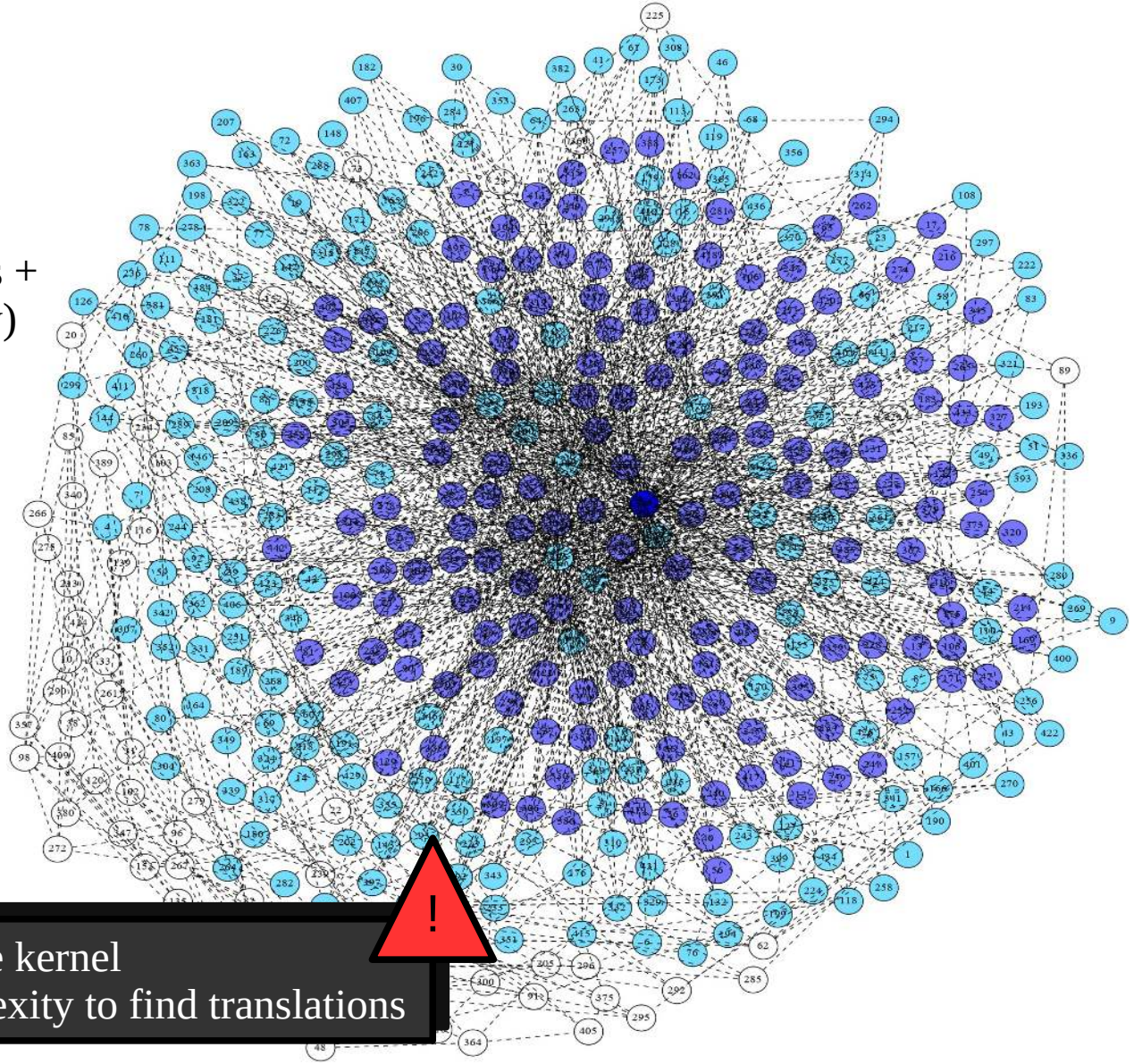
(smoothness + stationarity)



Results — The Haxby dataset

$\mathcal{G}_{\mathrm{K}}^{*}$

(smoothness + stationarity)

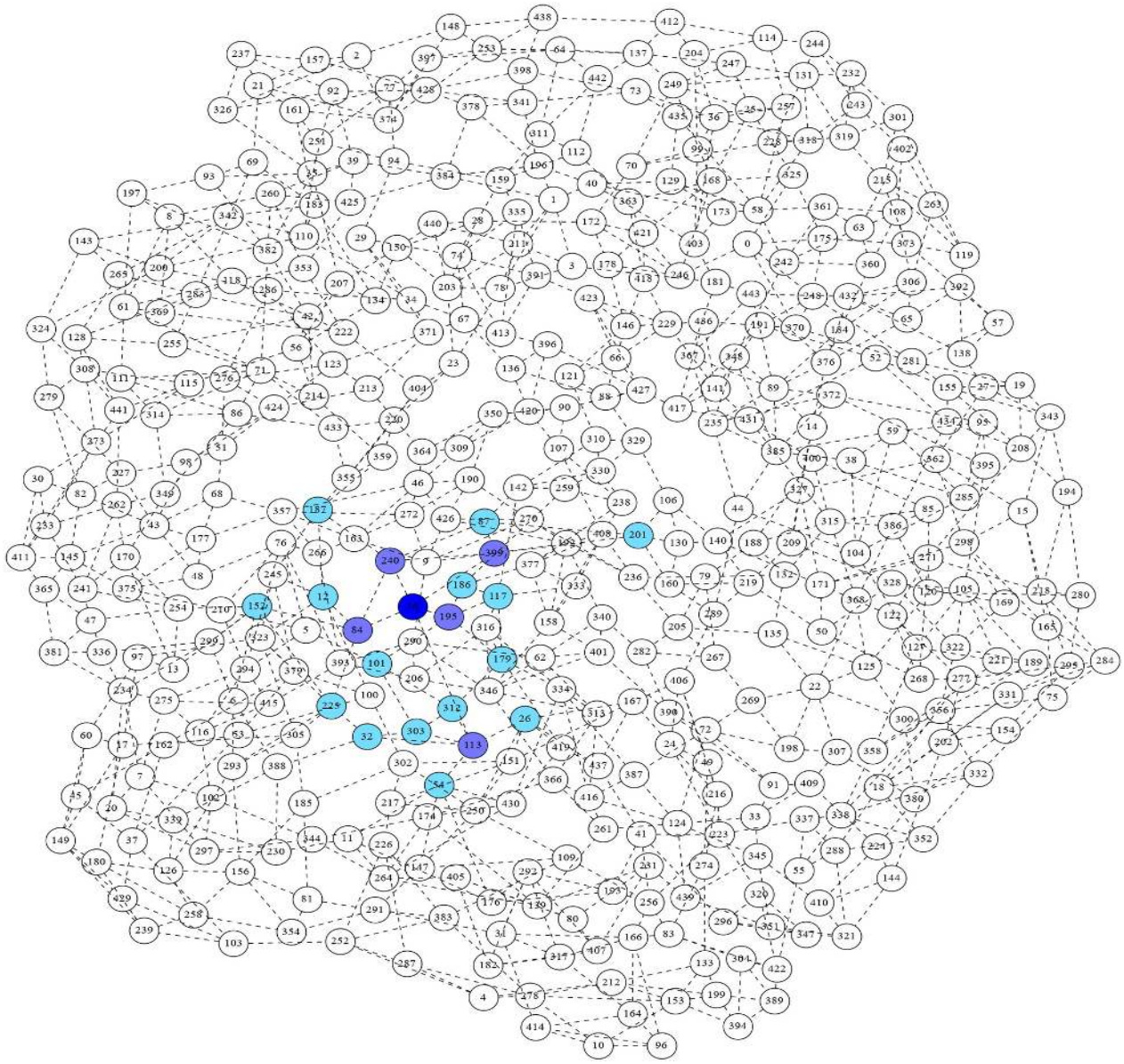


No locality of the kernel
Very high complexity to find translations

Results — The Haxby dataset

$\mathcal{G}_{\text{geo}}$

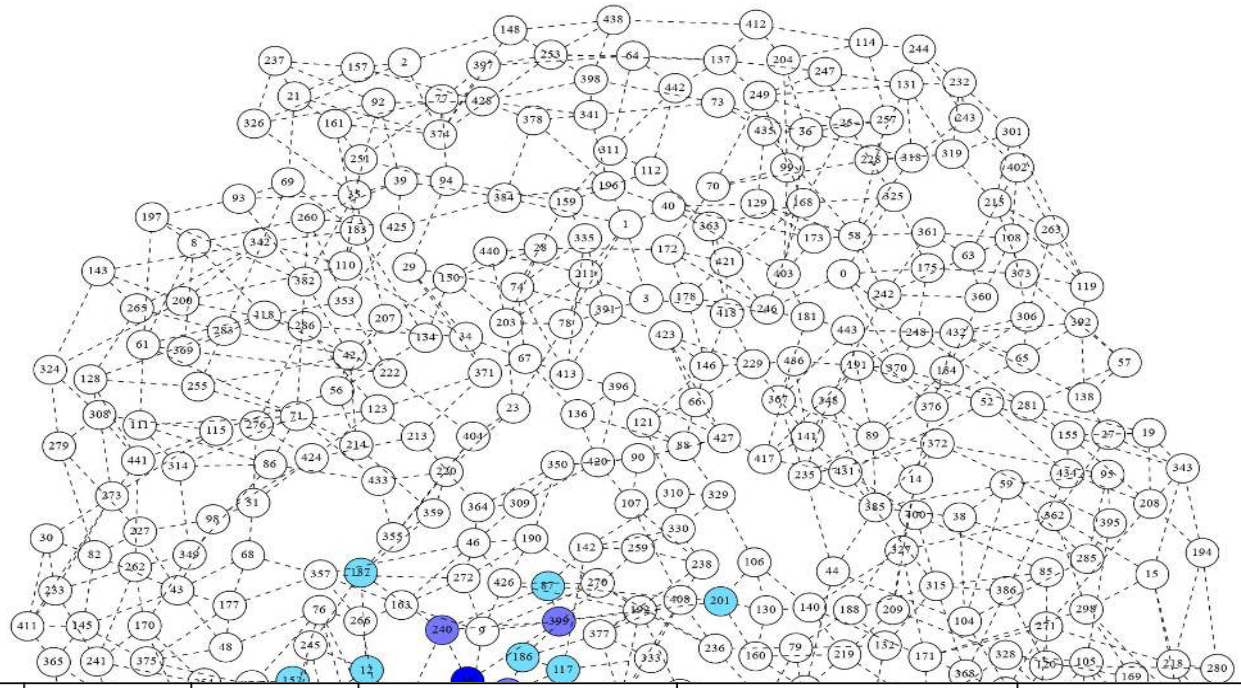
(geometric coordinates)



Results — The Haxby dataset

$\mathcal{G}_{\text{geo}}$

(geometric coordinates)

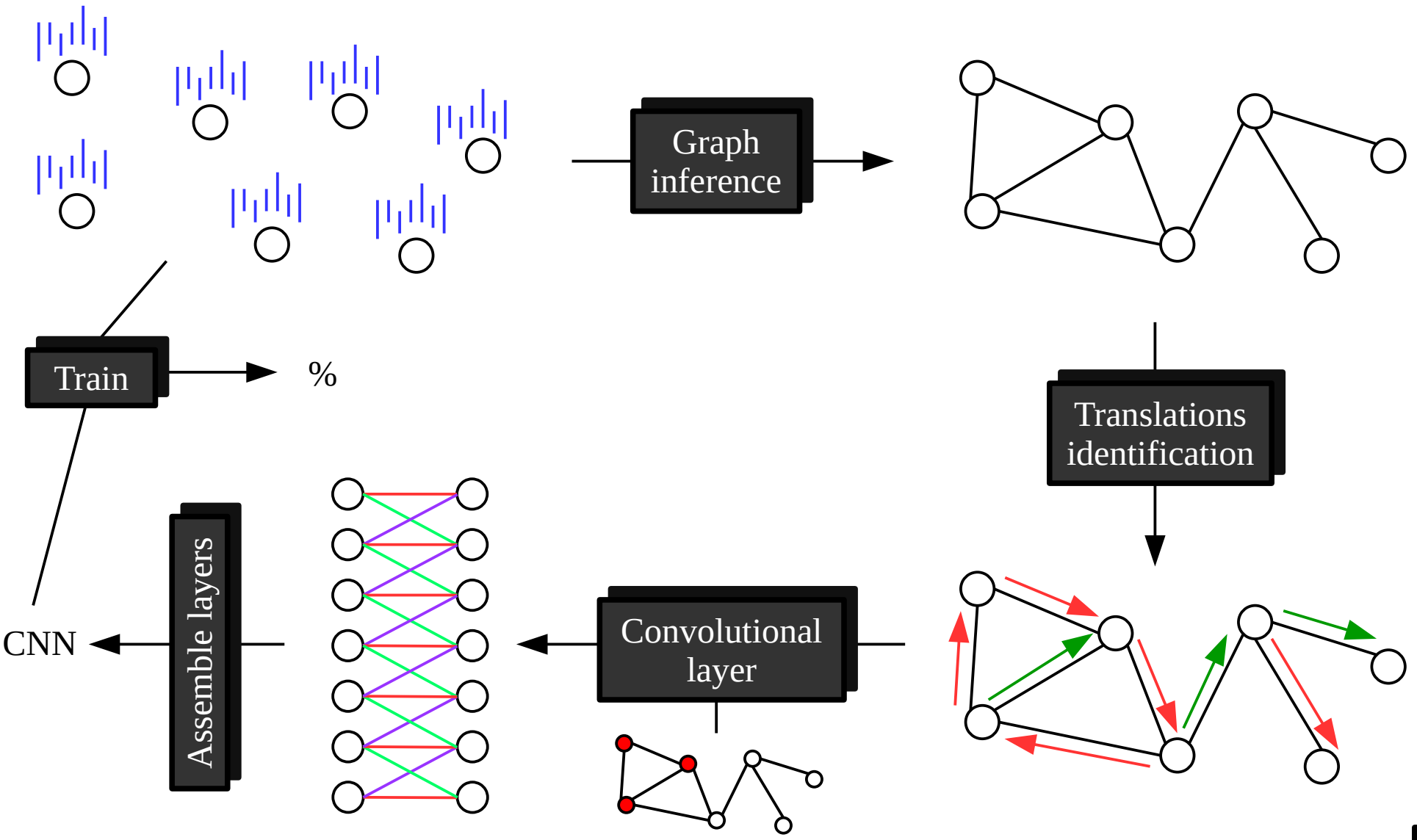


	$\mathcal{G}_{geo}$	MLP	SVM	LR ($C = 1, l_1$)	LR ($C = 50, l_1$)	(LR $C = 1, l_2$)	Ridge
Subject 1	**56%**	54%	50%	51%	47%	49%	35%
Subject 2	**52%**	44%	42%	48%	47%	47%	34%
Subject 3	47%	**48%**	35%	44%	38%	41%	28%
Subject 4	**53%**	42%	32%	34%	28%	34%	22%
Subject 5	**42%**	39%	31%	**42%**	34%	31%	23%
Subject 6	**49%**	**49%**	40%	42%	40%	38%	31%
Average	**49.9%**	45.9%	38.2%	43.6%	38.9%	39.7%	28.8%

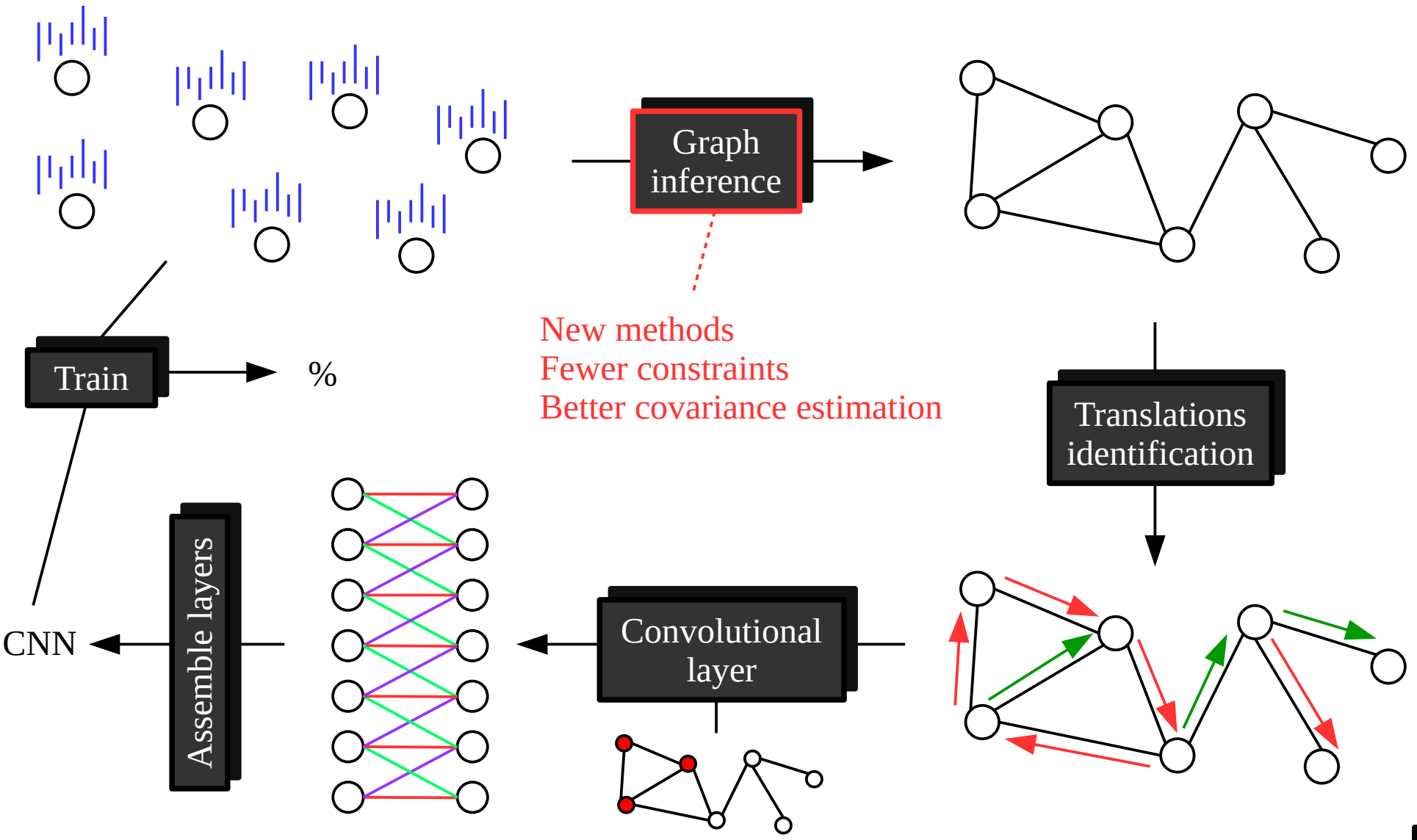
Outline

Conclusions & perspectives

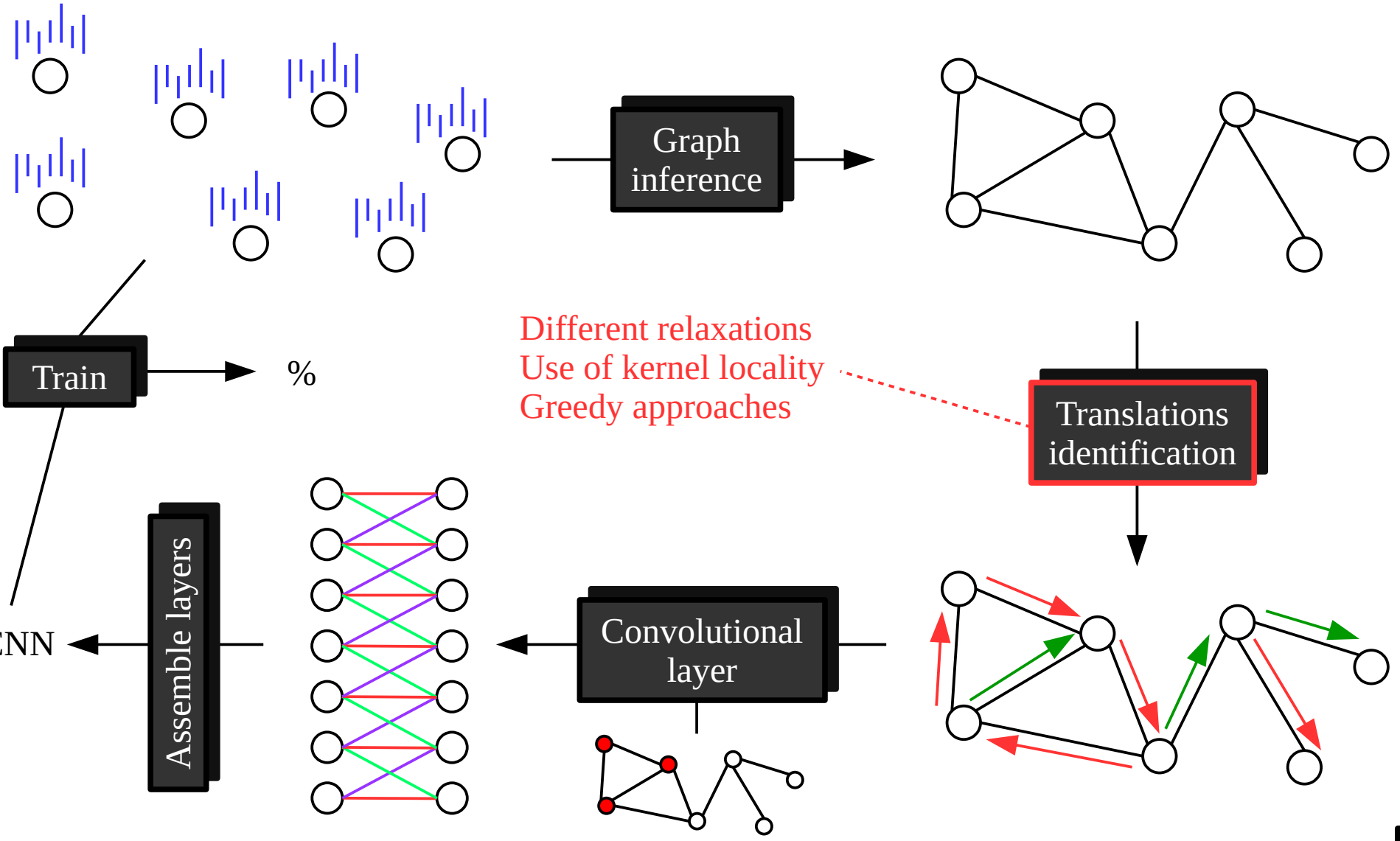
Conclusion & perspectives — Summary of the method



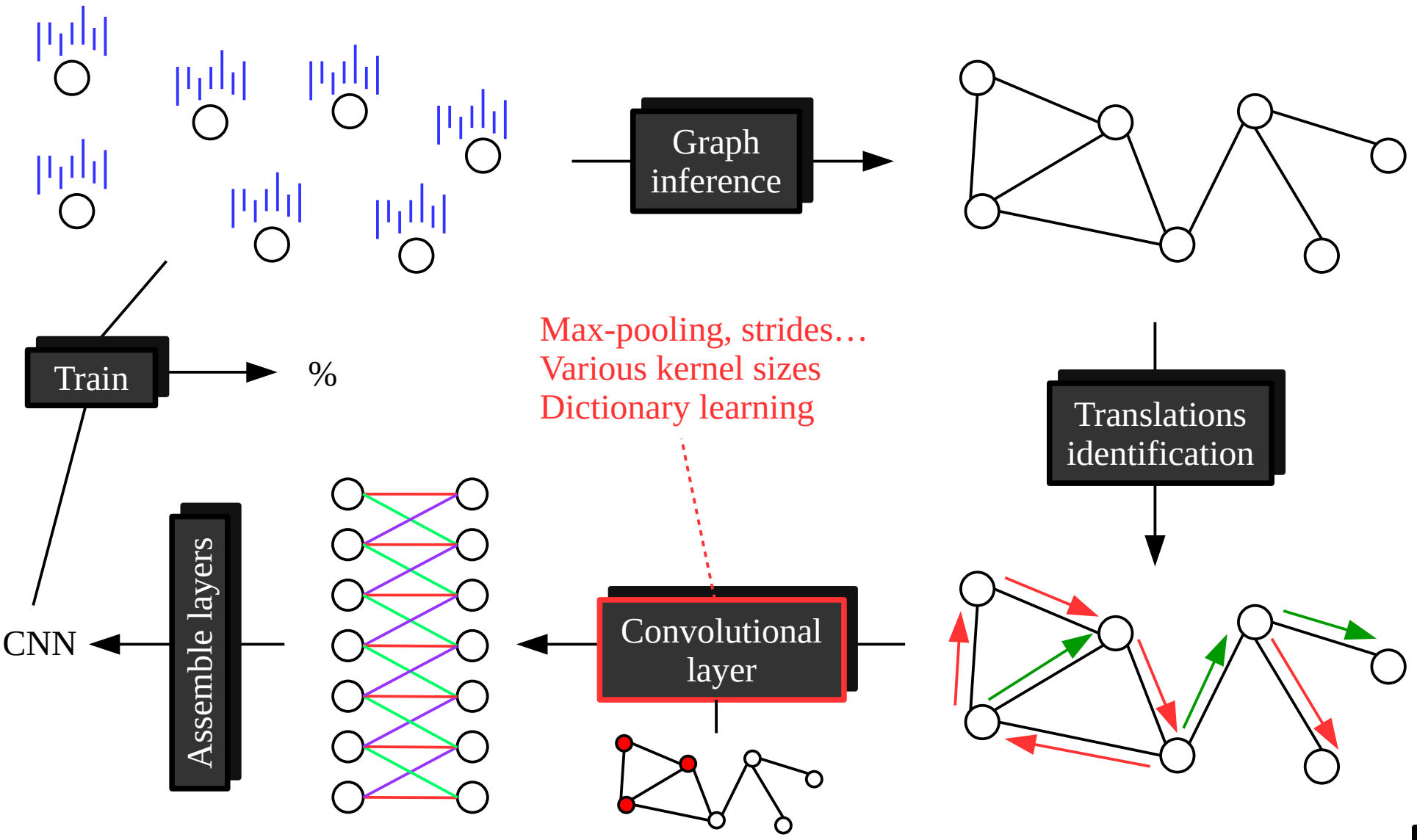
Conclusion & perspectives — Graph inference



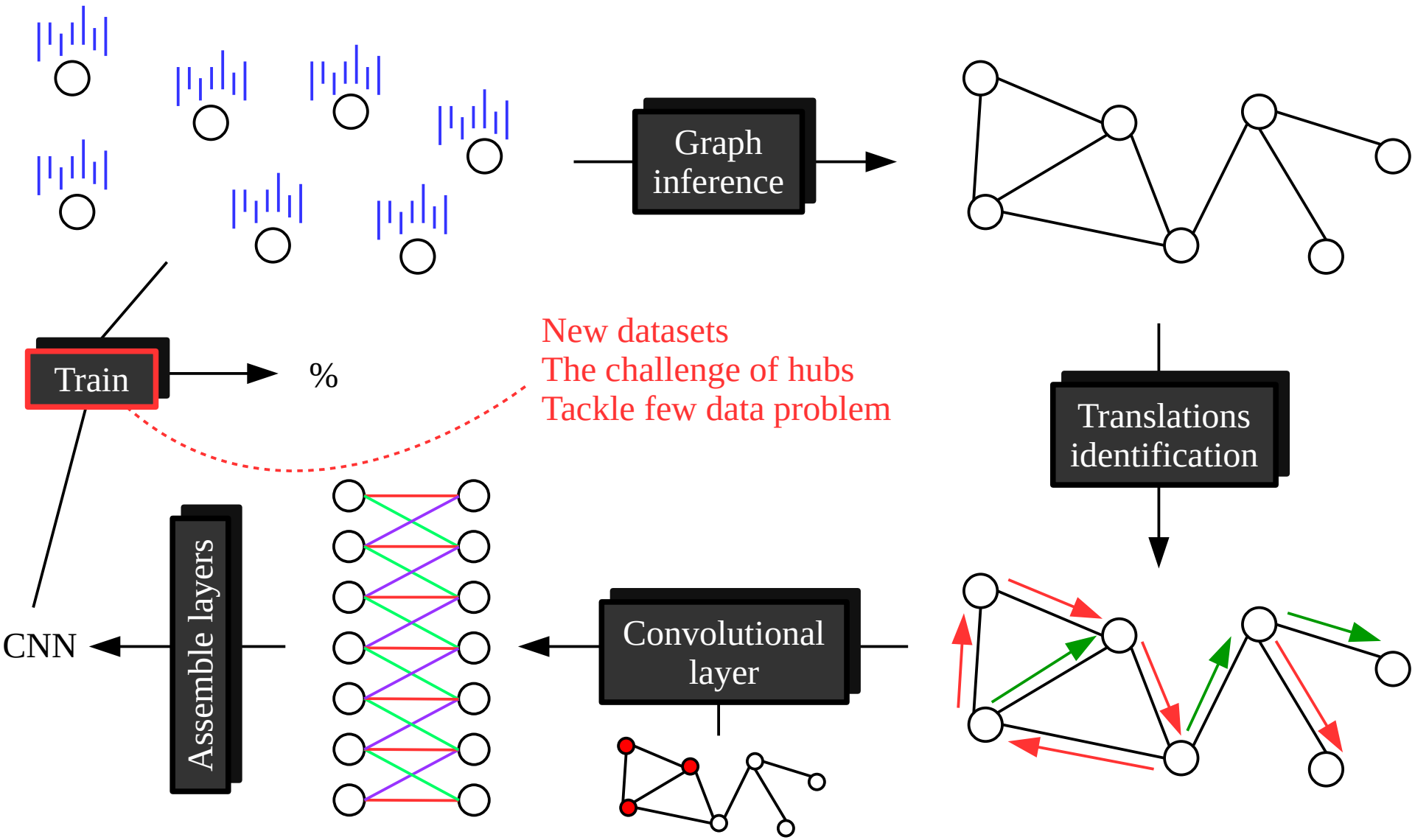
Conclusion & perspectives — Translations identification



Conclusion & perspectives — Convolutional layer



Conclusion & perspectives — CNN training



Conclusion & perpectives — Journals

Pasdeloup *et al.* — Translations on graphs with neighborhood preservation (IEEE Transactions on Signal Processing, Submitted)

Pasdeloup *et al.* — Characterization and inference of graph diffusion processes from observations of stationary signals (IEEE Transactions on Signal and Information Processing over Networks, 2017)

Conclusion & perpectives — Conferences

Pasdeloup *et al.* — Convolutional neural networks on irregular domains through approximate translations on inferred graphs (ICASSP, Submitted)

Ménoret *et al.* — Evaluating graph signal processing for neuroimaging through classification and dimensionality reduction (GlobalSIP, 2017)

Grelier *et al.* — Neighborhood-preserving translations on graphs (GlobalSIP, 2016)

Pasdeloup *et al.* — Towards a characterization of the uncertainty curve for graphs (ICASSP, 2016)

Pasdeloup *et al.* — Graph reconstruction from the observation of diffused signals (Allerton Conference on Communication, Control and Computing, 2015)

Pasdeloup *et al.* — Toward an uncertainty principle for weighted graphs (EUSIPCO, 2016)

Conclusion & perpectives — Workshops

Grelier *et al.* — Translation sur graphe avec préservation du voisinage (Gretsi, 2017)

Pasdeloup *et al.* — Characterization and inference of graph diffusion processes from observations of stationary processes (Bellairs workshop on graph signal processing, 2017)

Pasdeloup *et al.* — Characterizing graphs from diffused signals (Graph signal processing workshop, 2016)

Pasdeloup *et al.* — Characterizing graphs from diffused signals (Workshop on data driven approach to networks and languages, 2016)

Pasdeloup *et al.* — Vers une caractérisation de la courbe d’incertitude pour des graphes portant des signaux (Gretsi, 2015)

Conclusion & perpectives — Education

Conferences

Pasdeloup *et al.* — PyRat – Un jeu sérieux pour l'enseignement de l'informatique (QPES, 2017)

Workshops

Pasdeloup *et al.* — PyRat – Un jeu pour l'apprentissage de l'informatique avec Python (Journées UTICE, 2016)

Pasdeloup *et al.* — PyRat – Un jeu pour l'apprentissage de l'informatique avec Python (PyCon, 2016)



Extending Convolutional Neural Networks to Irregular Domains through Graph Inference

Bastien Pasdeloup

bastien.pasdeloup@imt-atlantique.fr

Ph.D. director: Dominique Pastor
Ph.D. advisor: Vincent Gripon

December 12, 2017

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Lab-STICC

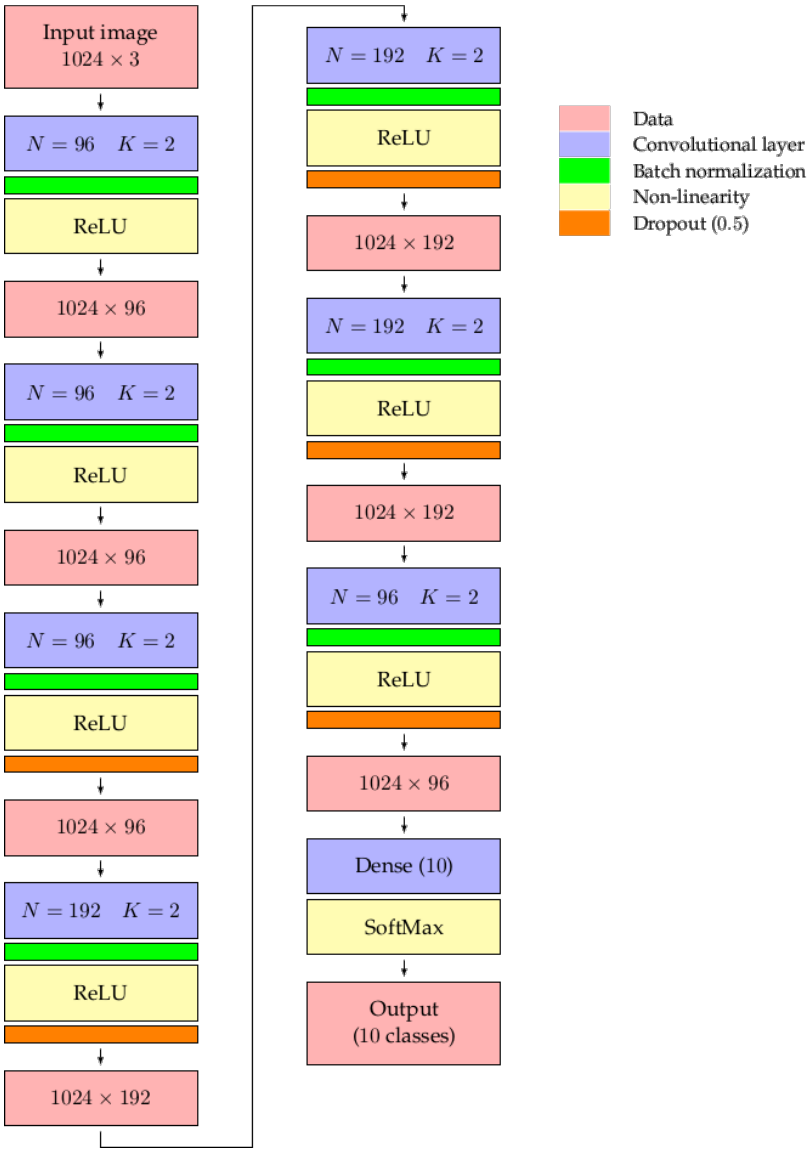
UNIVERSITE BRETAGNE LOIRE / MATHSTIC

Outline

Supplementary slides

— CNN architecture used for CIFAR-10 —

Supplementary — CNN architecture — Details



7 convolutional layers

1 kernel size (2 hops)

Dropout (avoid overfitting)

~~Max-pooling, strides…~~

Outline

Supplementary slides

— Optimization problem to approximate translations —

$$\mathbf{A}_\psi^* = \underset{\mathbf{A}_\psi \in \{0,1\}^{N \times N}}{\operatorname{argmin}} \left\| \mathbf{A}\mathbf{A}_\psi - \mathbf{A}_\psi \mathbf{A} \right\|_1$$

$$s.t. \quad \begin{cases} \forall i, j \in [\![1, N]\!] : \mathbf{A}_\psi[i, j] \leq \mathbf{A}[i, j] \\ \forall j \in [\![1, N]\!] : \sum_{i=1}^{N} \mathbf{A}_\psi[i, j] \leq 1 \\ \forall i \in [\![1, N]\!] : \sum_{j=1}^{N} \mathbf{A}_\psi[i, j] \leq 1 \\ \left\| \mathbf{A}_\psi \right\|_1 = C \end{cases}$$

$$\mathbf{A}_\psi^* = \underset{\mathbf{A}_\psi \in \{0,1\}^{N \times N}}{\operatorname{argmin}} \left\| \mathbf{A}\mathbf{A}_\psi - \mathbf{A}_\psi \mathbf{A} \right\|_1$$

Minimize SNP violations

$$s.t. \begin{cases} \forall i,j \in [\![1,N]\!] : \mathbf{A}_\psi[i,j] \leq \mathbf{A}[i,j] \\ \forall j \in [\![1,N]\!] : \sum_{i=1}^{N} \mathbf{A}_\psi[i,j] \leq 1 \\ \forall i \in [\![1,N]\!] : \sum_{j=1}^{N} \mathbf{A}_\psi[i,j] \leq 1 \\ \left\| \mathbf{A}_\psi \right\|_1 = C \end{cases}$$

EC

Injective

Scale factor

$$\mathbf{A}_\psi^* = \underset{\mathbf{A}_\psi \in \{0,1\}^{N \times N}}{\operatorname{argmin}} \|\mathbf{A}\mathbf{A}_\psi - \mathbf{A}_\psi\mathbf{A}\|_1 + \alpha\|\mathbf{A} - \mathbf{A}_\psi\|_1$$

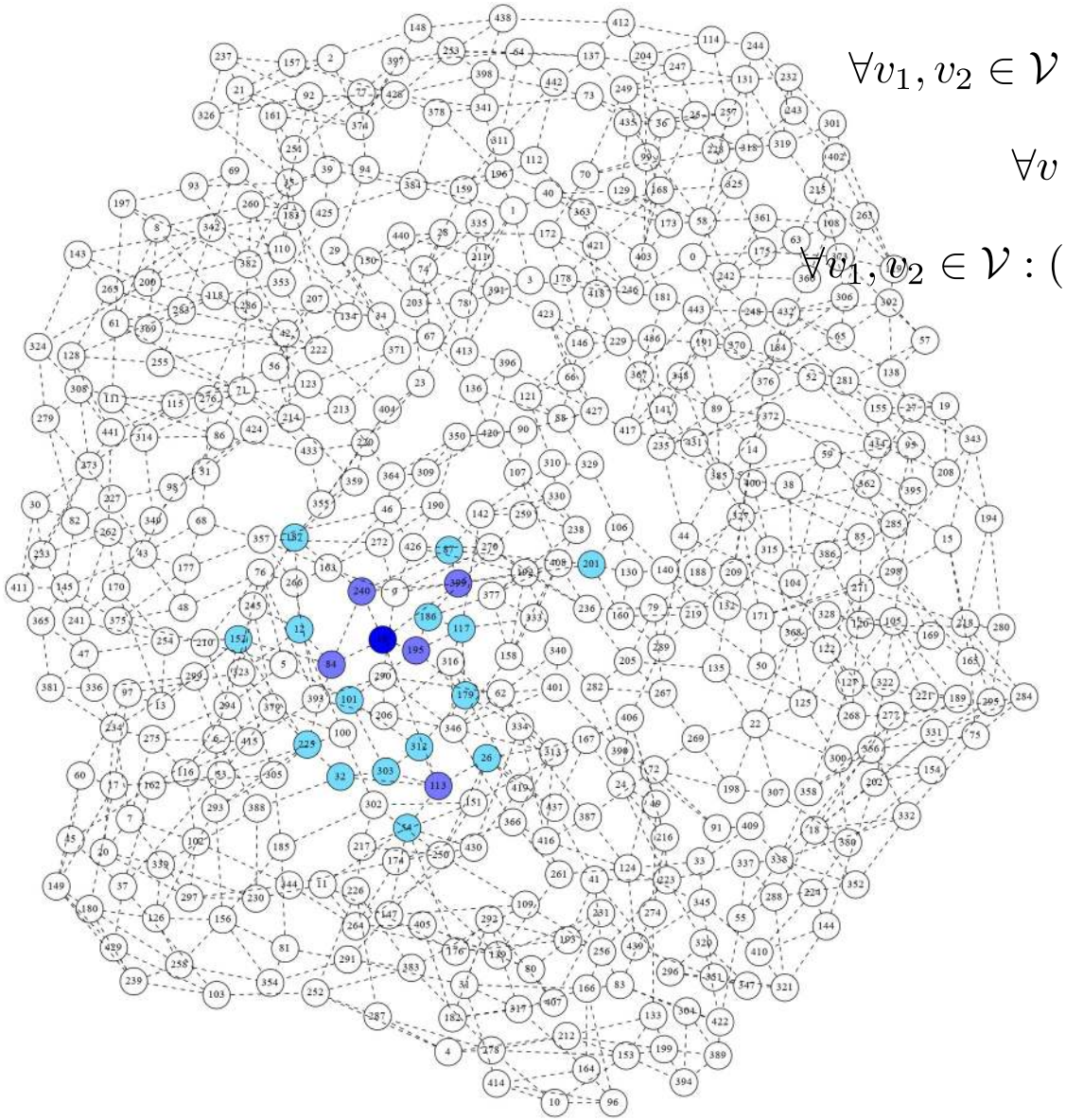
$$s.t. \begin{cases} \forall j \in [\![1, N]\!] : \sum_{i=1}^{N} \mathbf{A}_\psi[i,j] \leq 1 \\ \forall i \in [\![1, N]\!] : \sum_{j=1}^{N} \mathbf{A}_\psi[i,j] \leq 1 \\ \|\mathbf{A}_\psi\|_1 = C \end{cases},$$

Outline

Supplementary slides

— ICASSP 2018 submission —

Supplementary — ICASSP 2018 — Limits of our definitions

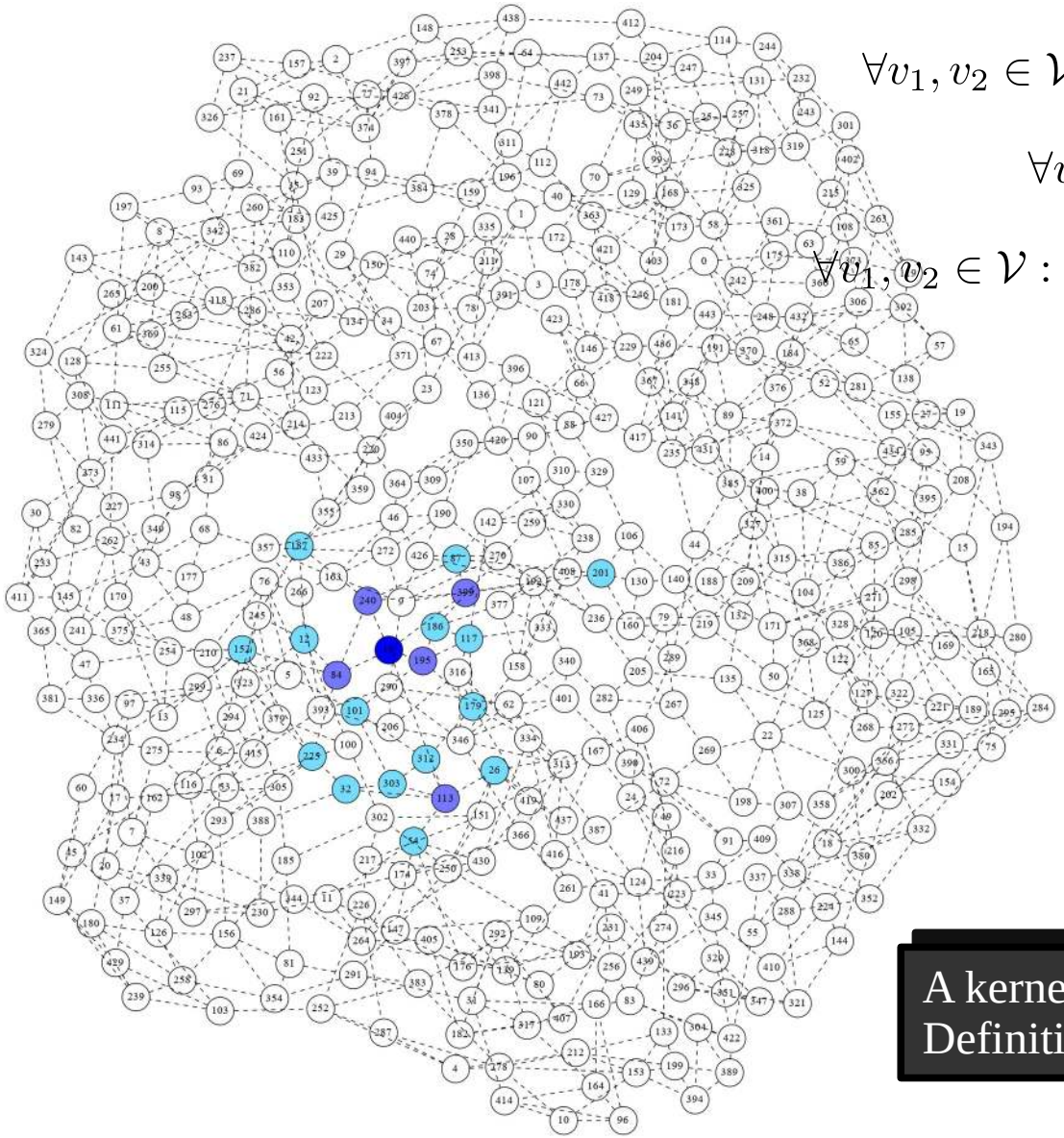


$$\forall v_1, v_2 \in \mathcal{V} : (\psi(v_1) = \psi(v_2) \neq \perp) \Rightarrow (v_1 = v_2)$$

$$\forall v \in \mathcal{V} : (\{v, \psi(v)\} \in \mathcal{E}) \vee (\psi(v) = \perp)$$

$$\forall v_1, v_2 \in \mathcal{V} : (\{v_1, v_2\} \in \mathcal{E} \Leftrightarrow \{\psi(v_1), \psi(v_2)\} \in \mathcal{E}) \vee (\psi(v_1) = \perp) \vee (\psi(v_2) = \perp)$$

Supplementary — ICASSP 2018 — Limits of our definitions



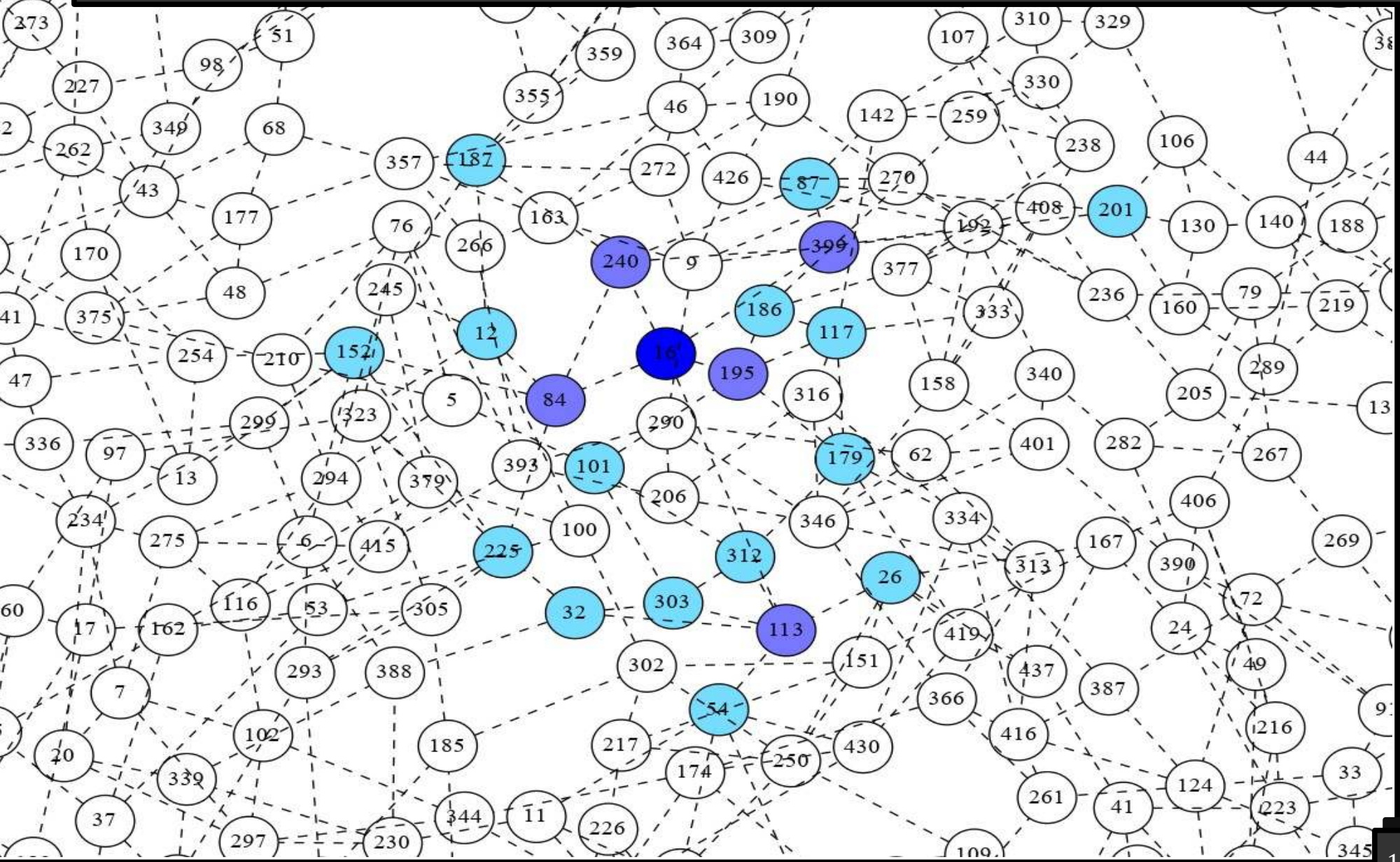
$$\forall v_1, v_2 \in \mathcal{V} : (\psi(v_1) = \psi(v_2) \neq \perp) \Rightarrow (v_1 = v_2)$$

$$\forall v \in \mathcal{V} : (\{v, \psi(v)\} \in \mathcal{E}) \vee (\psi(v) = \perp)$$

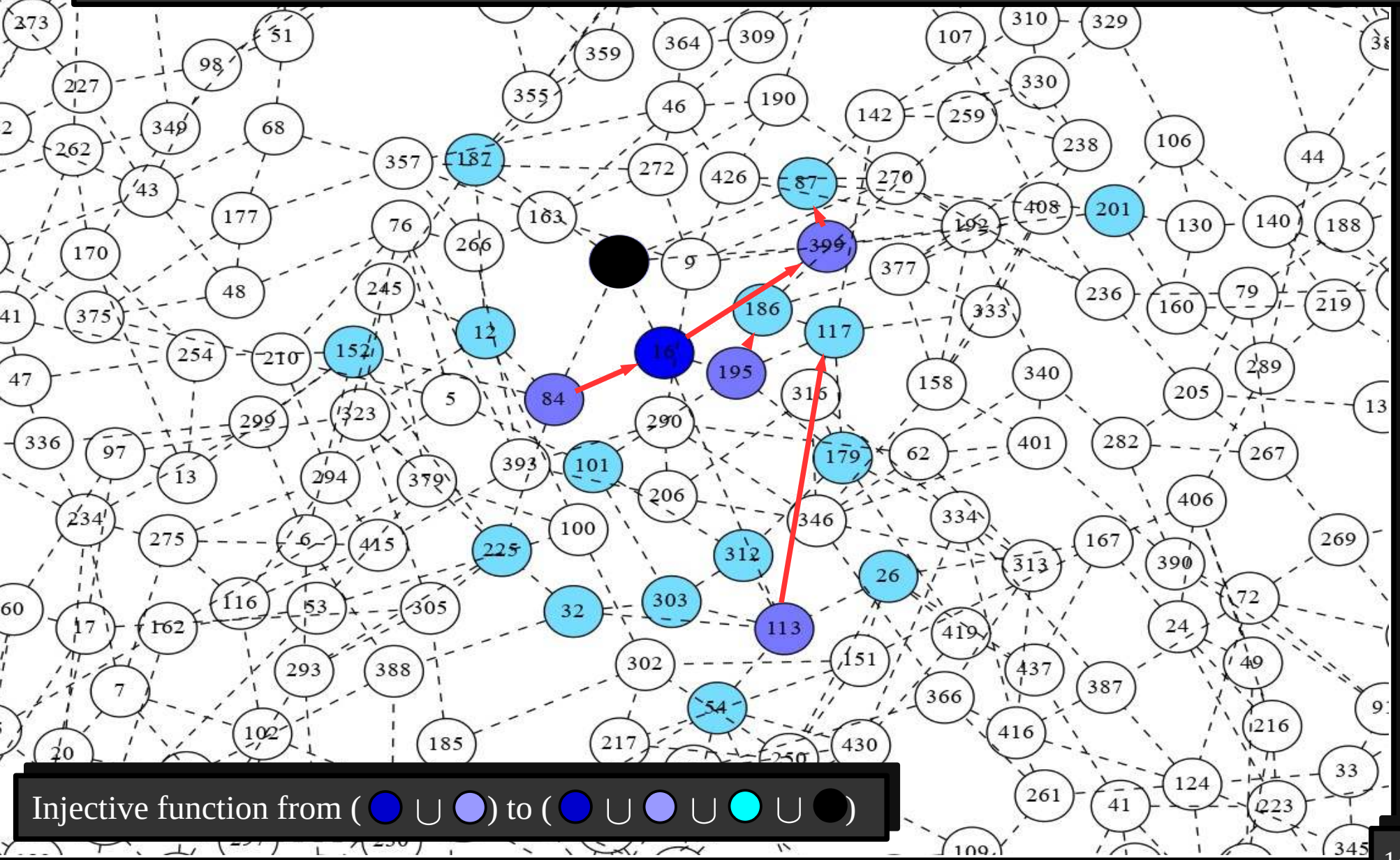
$$\forall v_1, v_2 \in \mathcal{V} : (\{v_1, v_2\} \in \mathcal{E} \Leftrightarrow \{\psi(v_1), \psi(v_2)\} \in \mathcal{E}) \vee (\psi(v_1) = \perp) \vee (\psi(v_2) = \perp)$$

A kernel is a localized signal
Definition of translations is very strict

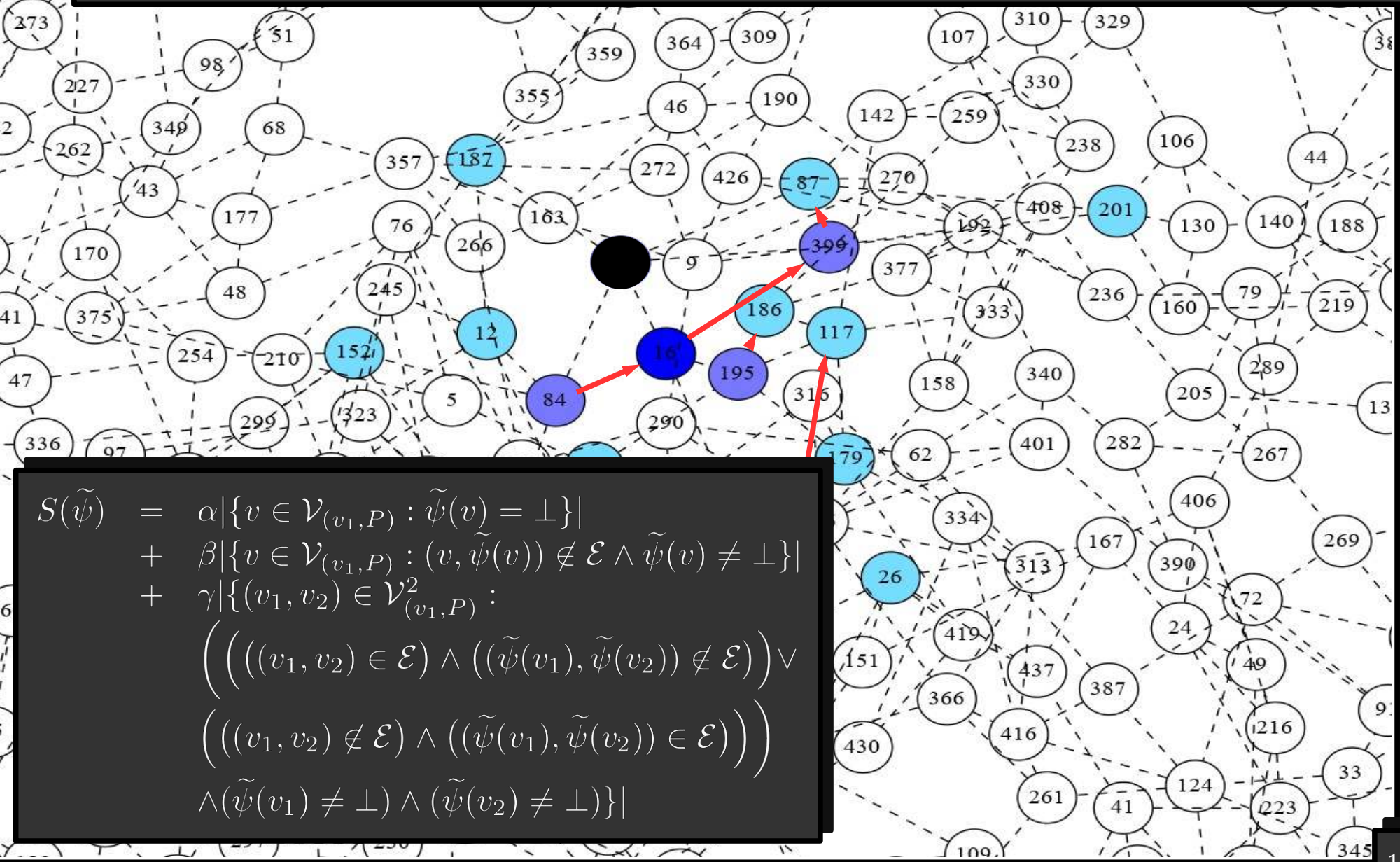
Supplementary — ICASSP 2018 — A local approach



Supplementary — ICASSP 2018 — A local approach

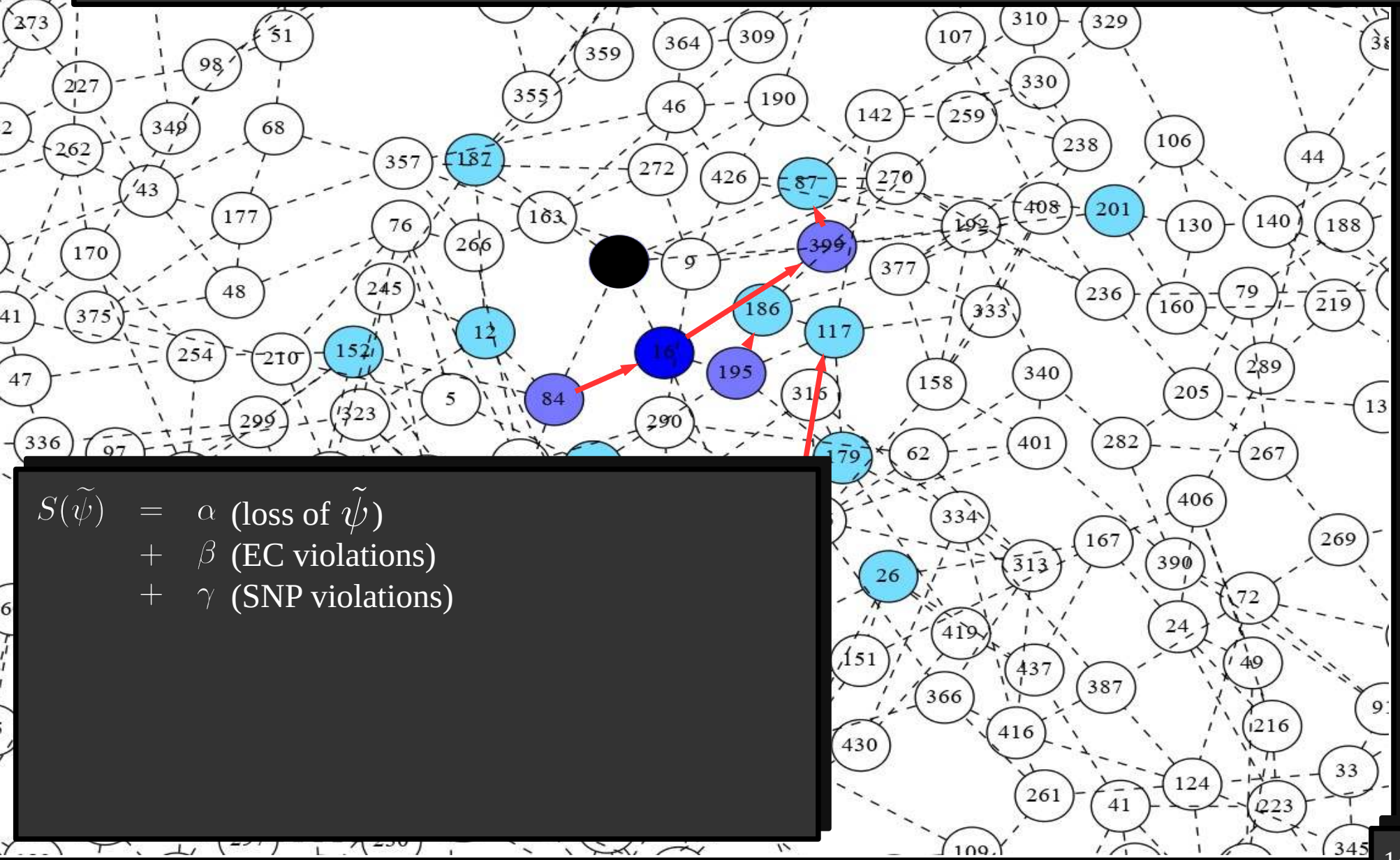


Supplementary — ICASSP 2018 — A local approach



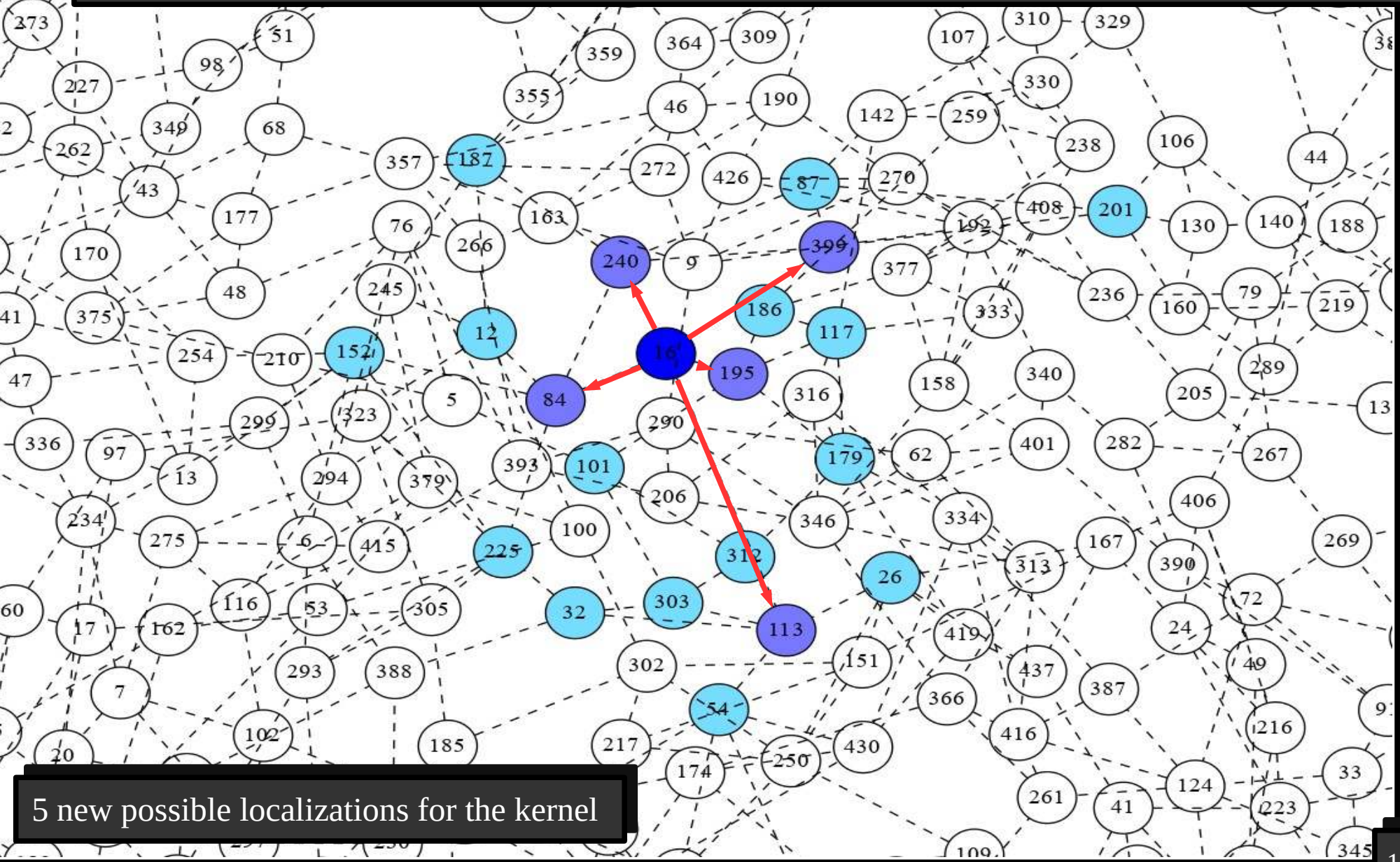
$$
\begin{aligned}
S(\widetilde{\psi}) \quad = \quad & \alpha|\{v \in \mathcal{V}_{(v_1,P)} : \widetilde{\psi}(v) = \perp\}| \\
+ \quad & \beta|\{v \in \mathcal{V}_{(v_1,P)} : (v, \widetilde{\psi}(v)) \notin \mathcal{E} \wedge \widetilde{\psi}(v) \neq \perp\}| \\
+ \quad & \gamma|\{(v_1, v_2) \in \mathcal{V}^2_{(v_1,P)} : \\
& \Big(\big((v_1, v_2) \in \mathcal{E}\big) \wedge \big((\widetilde{\psi}(v_1), \widetilde{\psi}(v_2)) \notin \mathcal{E}\big)\Big) \vee \\
& \Big(\big((v_1, v_2) \notin \mathcal{E}\big) \wedge \big((\widetilde{\psi}(v_1), \widetilde{\psi}(v_2)) \in \mathcal{E}\big)\Big) \Big) \\
& \wedge (\widetilde{\psi}(v_1) \neq \perp) \wedge (\widetilde{\psi}(v_2) \neq \perp)\}|
\end{aligned}
$$

Supplementary — ICASSP 2018 — A local approach



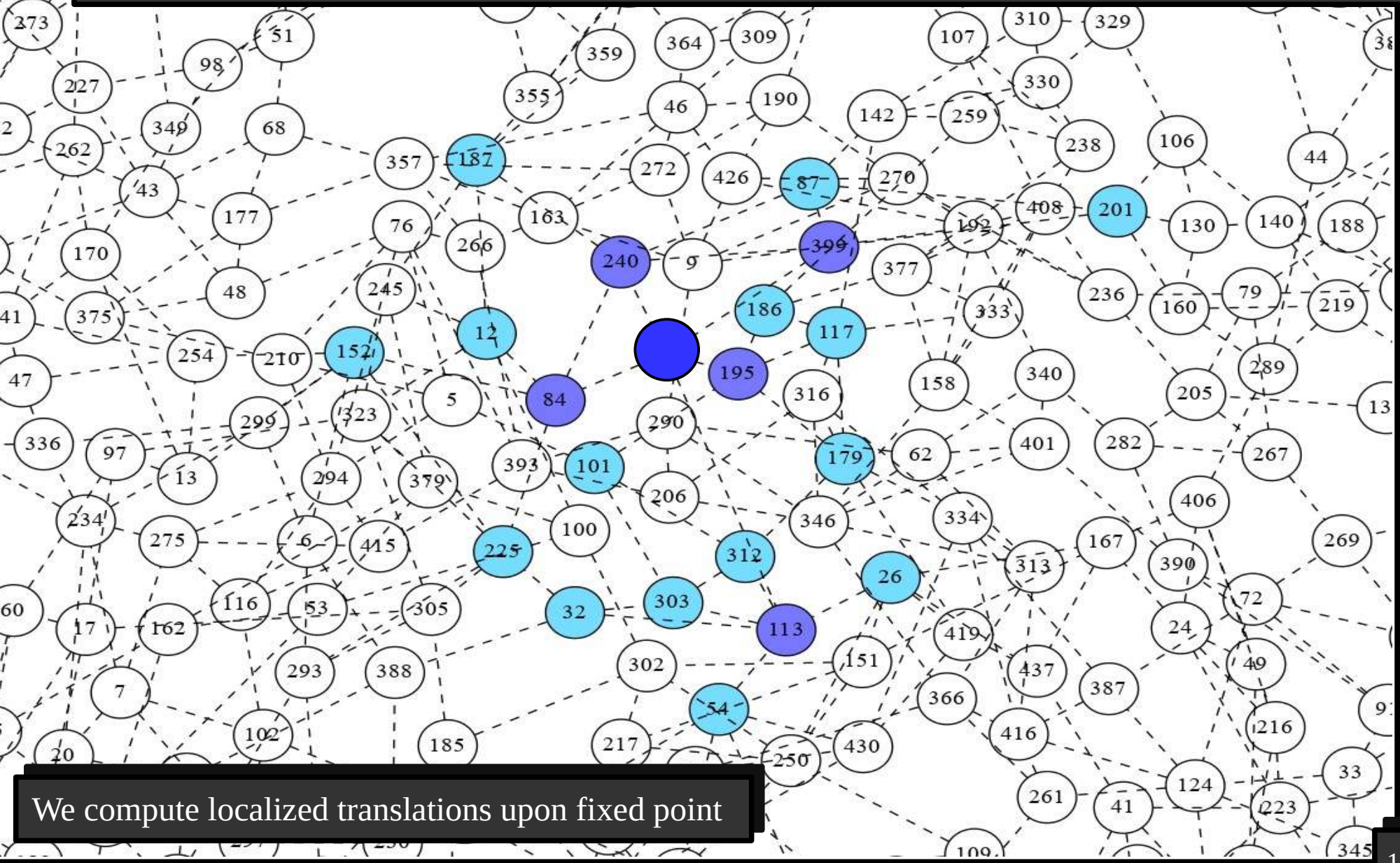
$$S(\tilde{\psi}) = \alpha \text{ (loss of } \tilde{\psi}\text{)} + \beta \text{ (EC violations)} + \gamma \text{ (SNP violations)}$$

Supplementary — ICASSP 2018 — A local approach



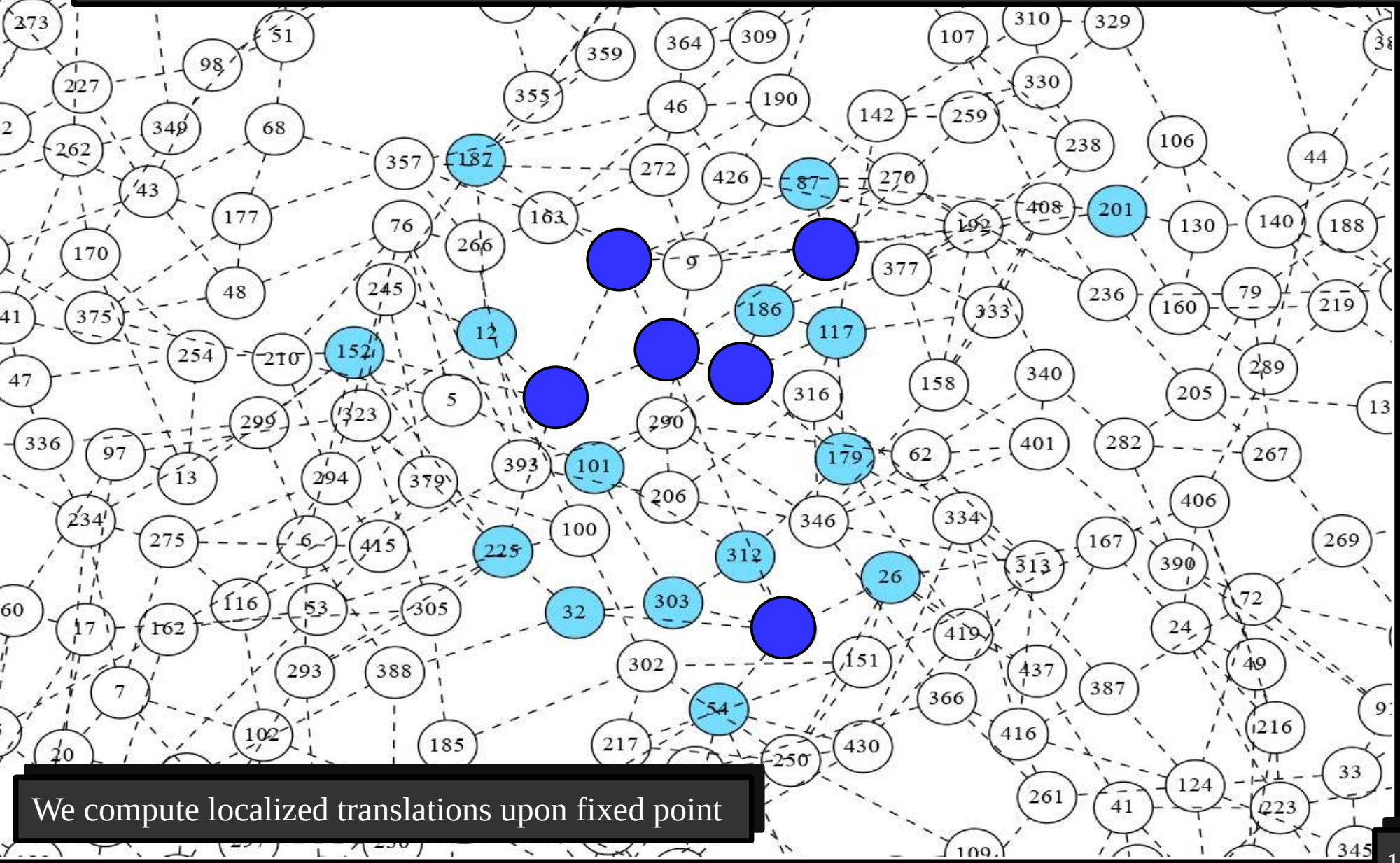
5 new possible localizations for the kernel

Supplementary — ICASSP 2018 — A local approach



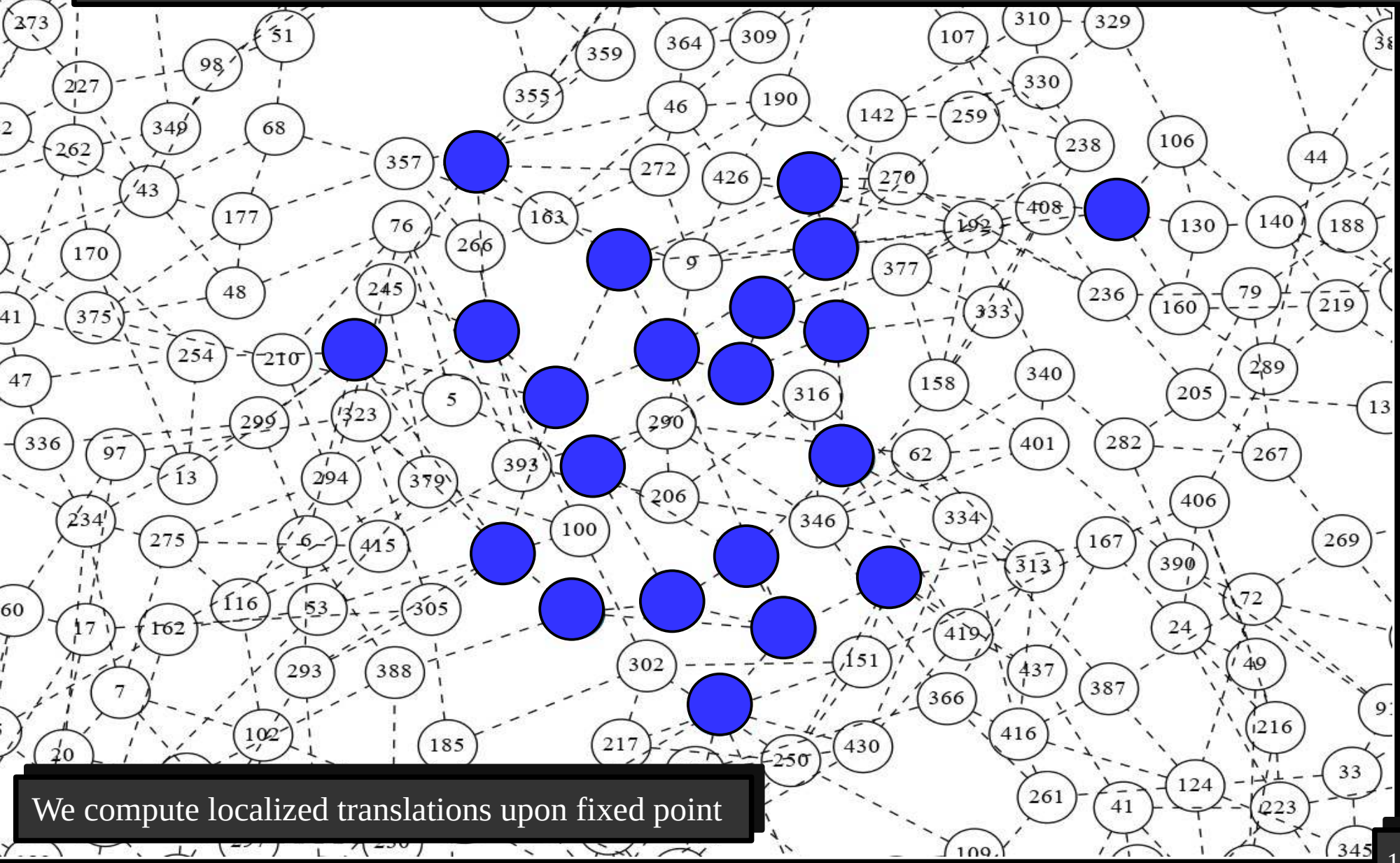
We compute localized translations upon fixed point

Supplementary — ICASSP 2018 — A local approach



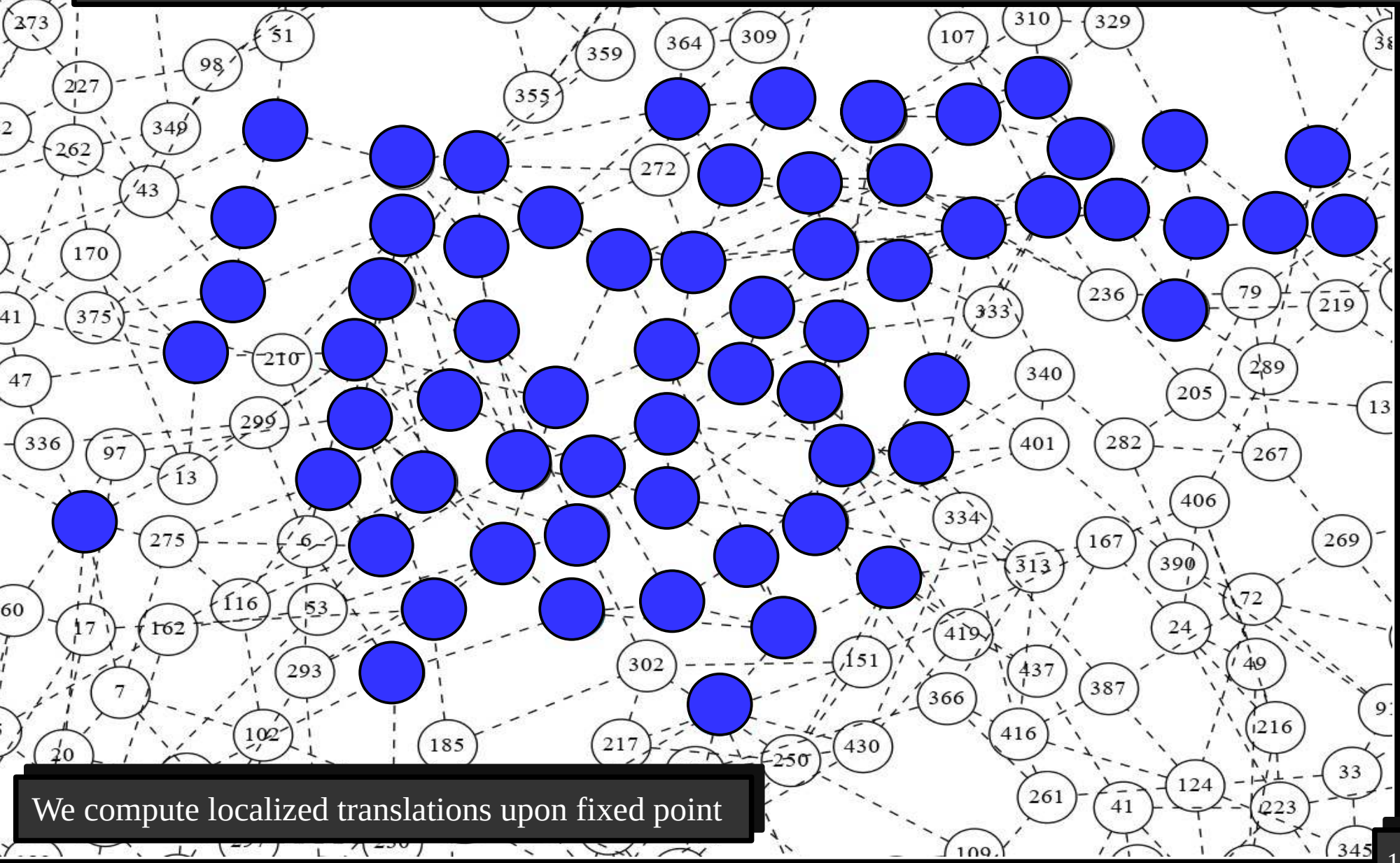
We compute localized translations upon fixed point

Supplementary — ICASSP 2018 — A local approach



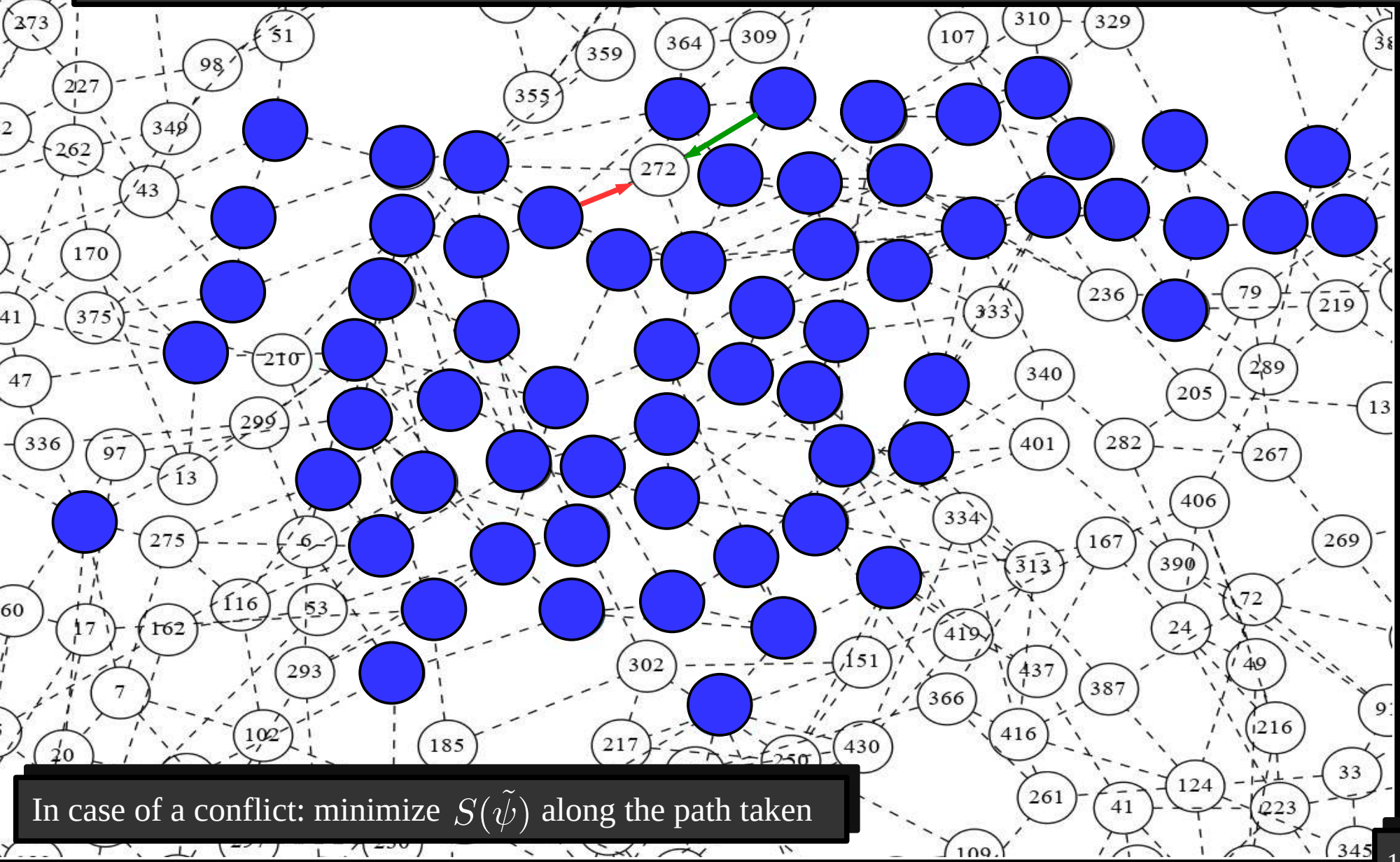
We compute localized translations upon fixed point

Supplementary — ICASSP 2018 — A local approach



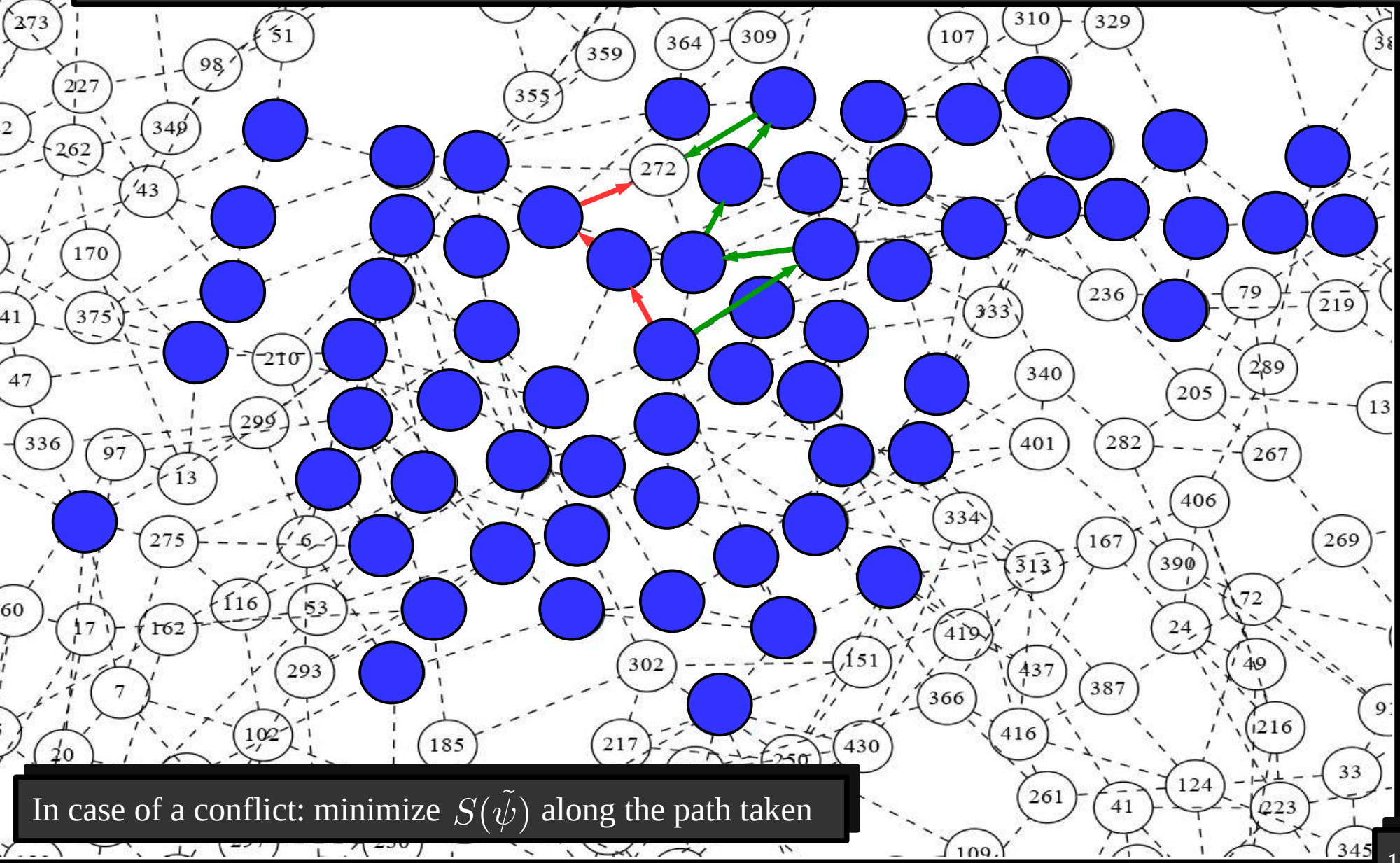
We compute localized translations upon fixed point

Supplementary — ICASSP 2018 — A local approach



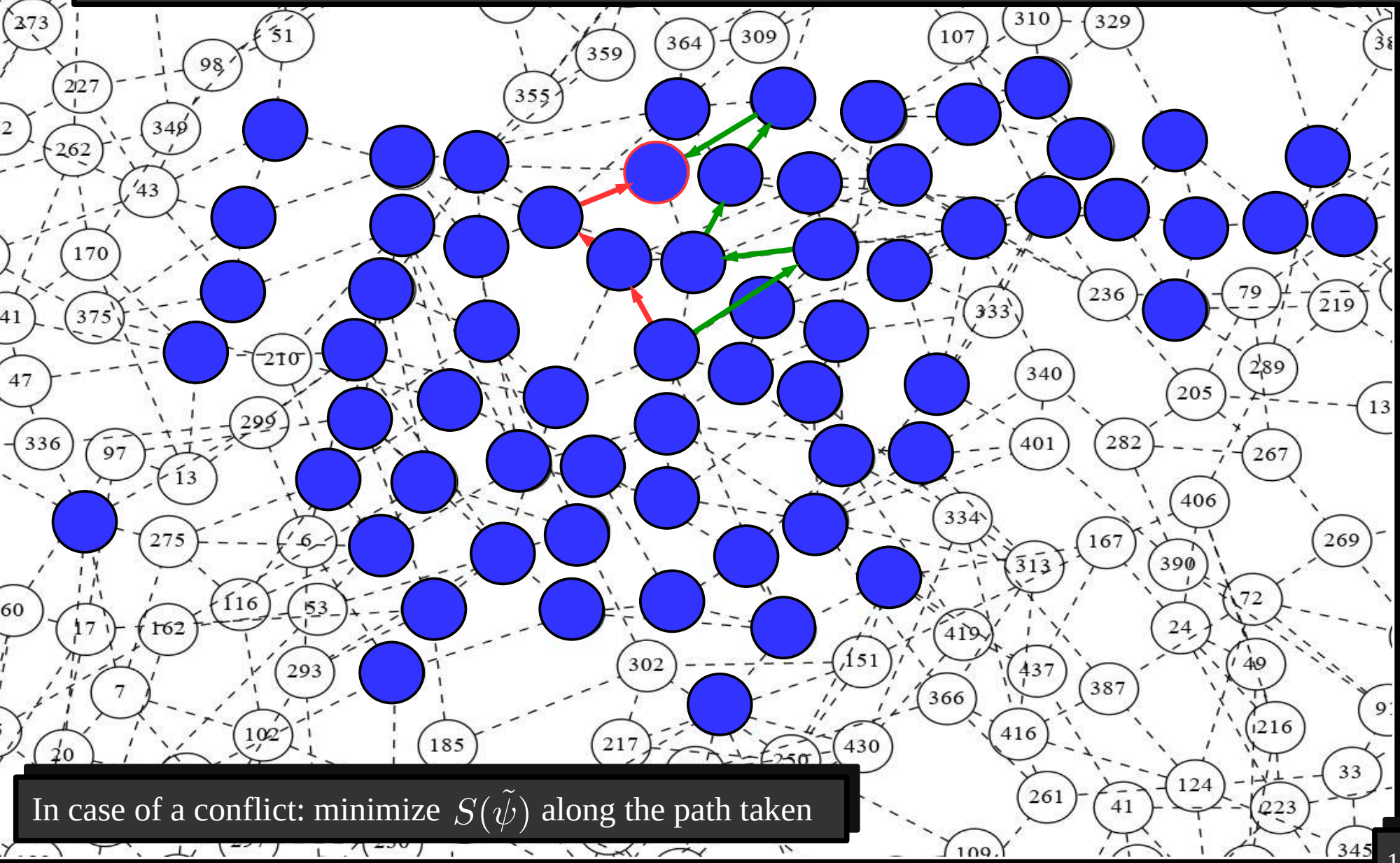
In case of a conflict: minimize $S(\tilde{\psi})$ along the path taken

Supplementary — ICASSP 2018 — A local approach



In case of a conflict: minimize $S(\tilde{\psi})$ along the path taken

Supplementary — ICASSP 2018 — A local approach



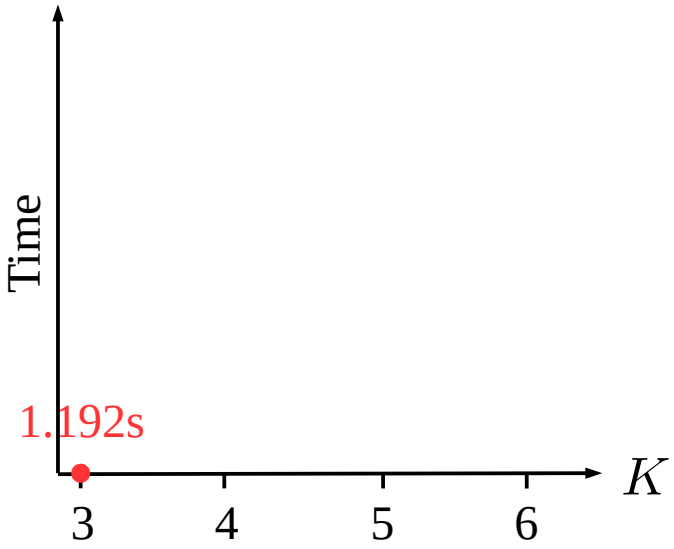
In case of a conflict: minimize $S(\tilde{\psi})$ along the path taken

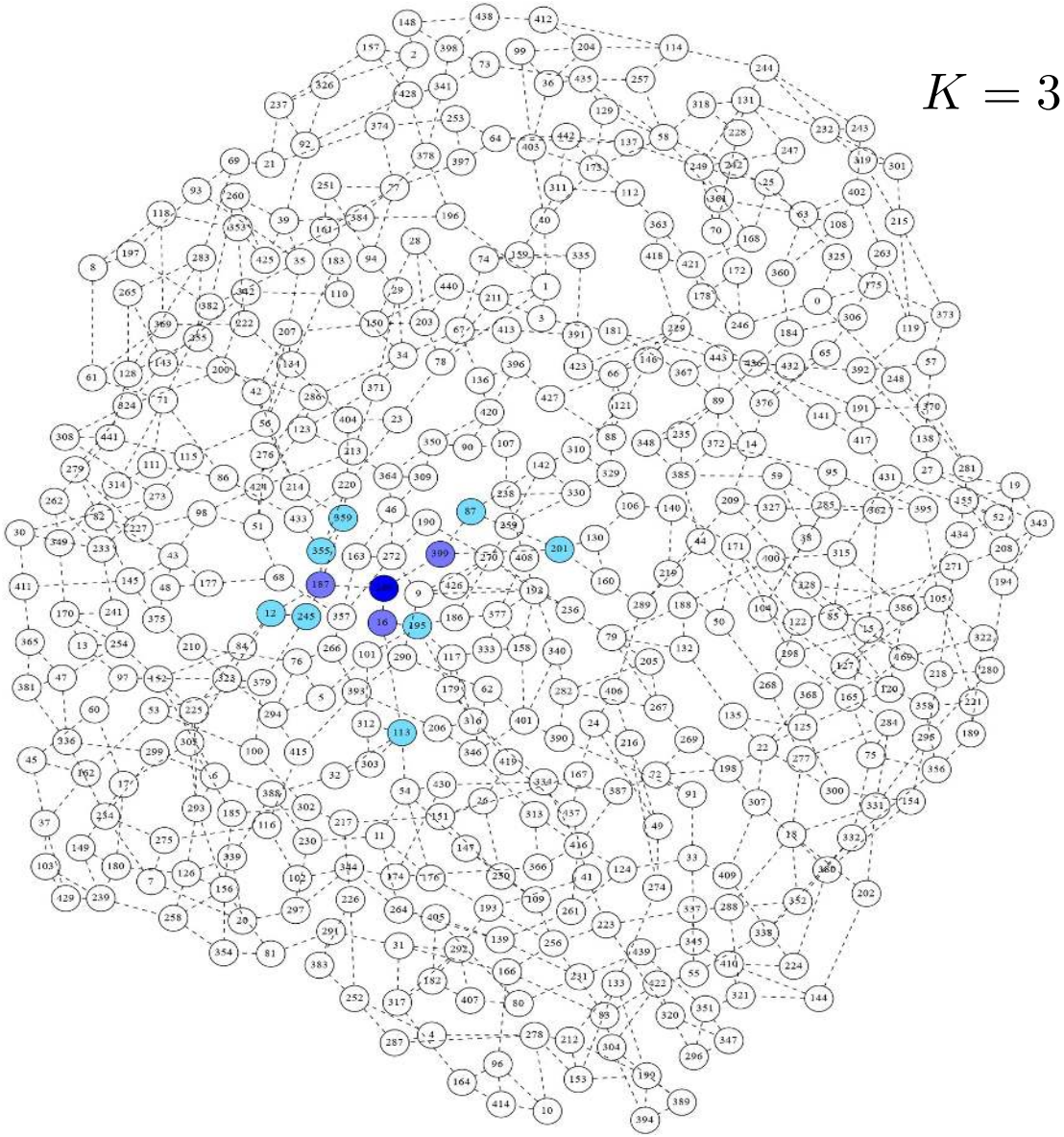
Outline

Supplementary slides

— Time measurements on Haxby —

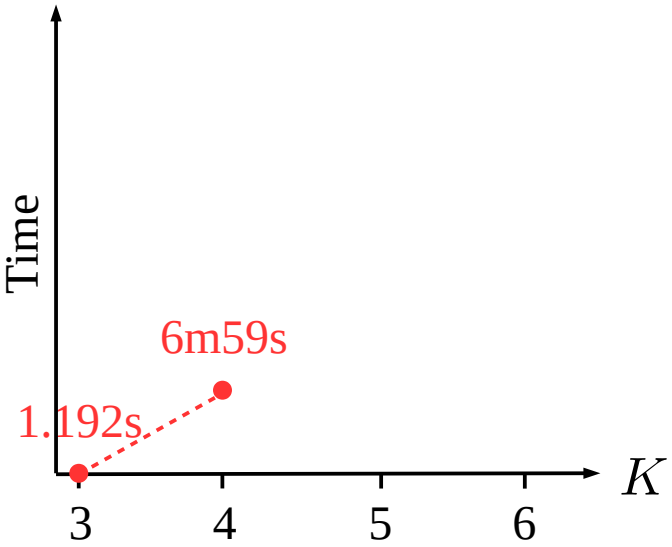
Supplementary — Time measurements — Function of K

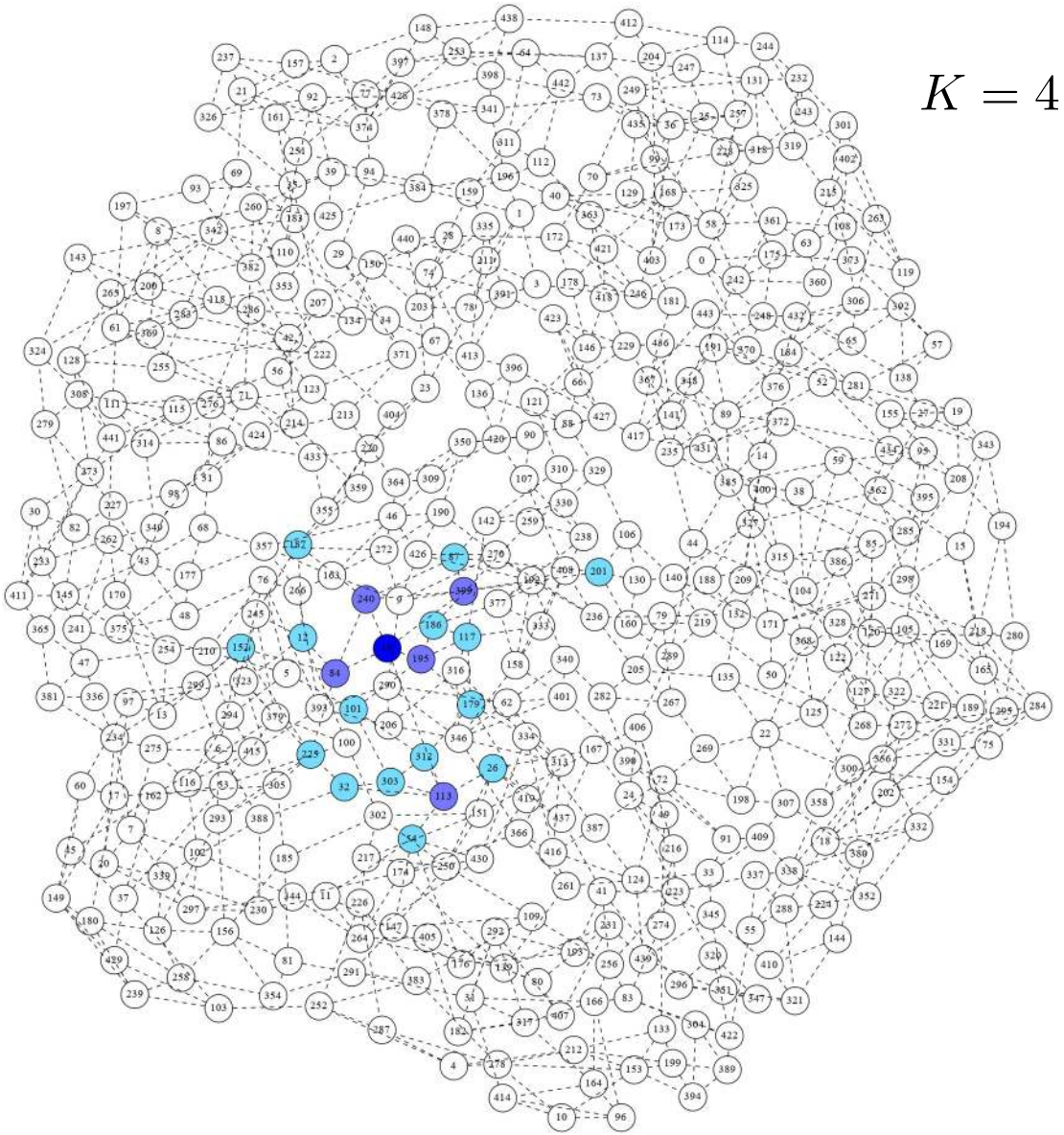




$\mathcal{G}_{\text{geo}}, \alpha = 1.0, \beta = 0.7, \gamma = 0.5$
444 vertices
C++ implementation w/ backtracking

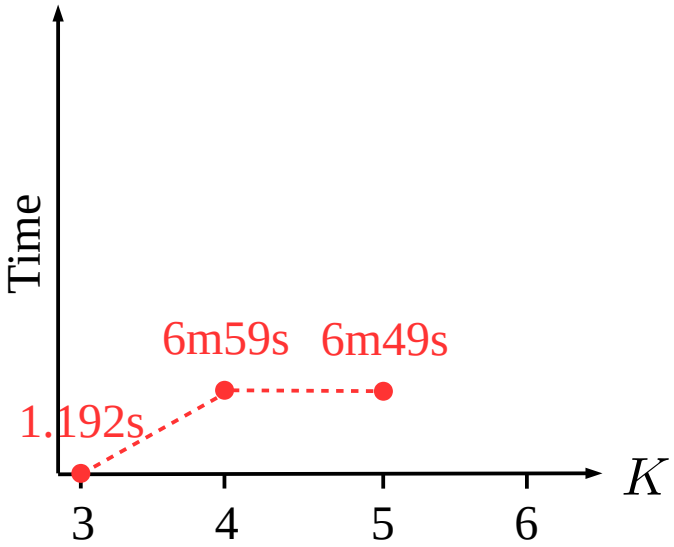
Supplementary — Time measurements — Function of K

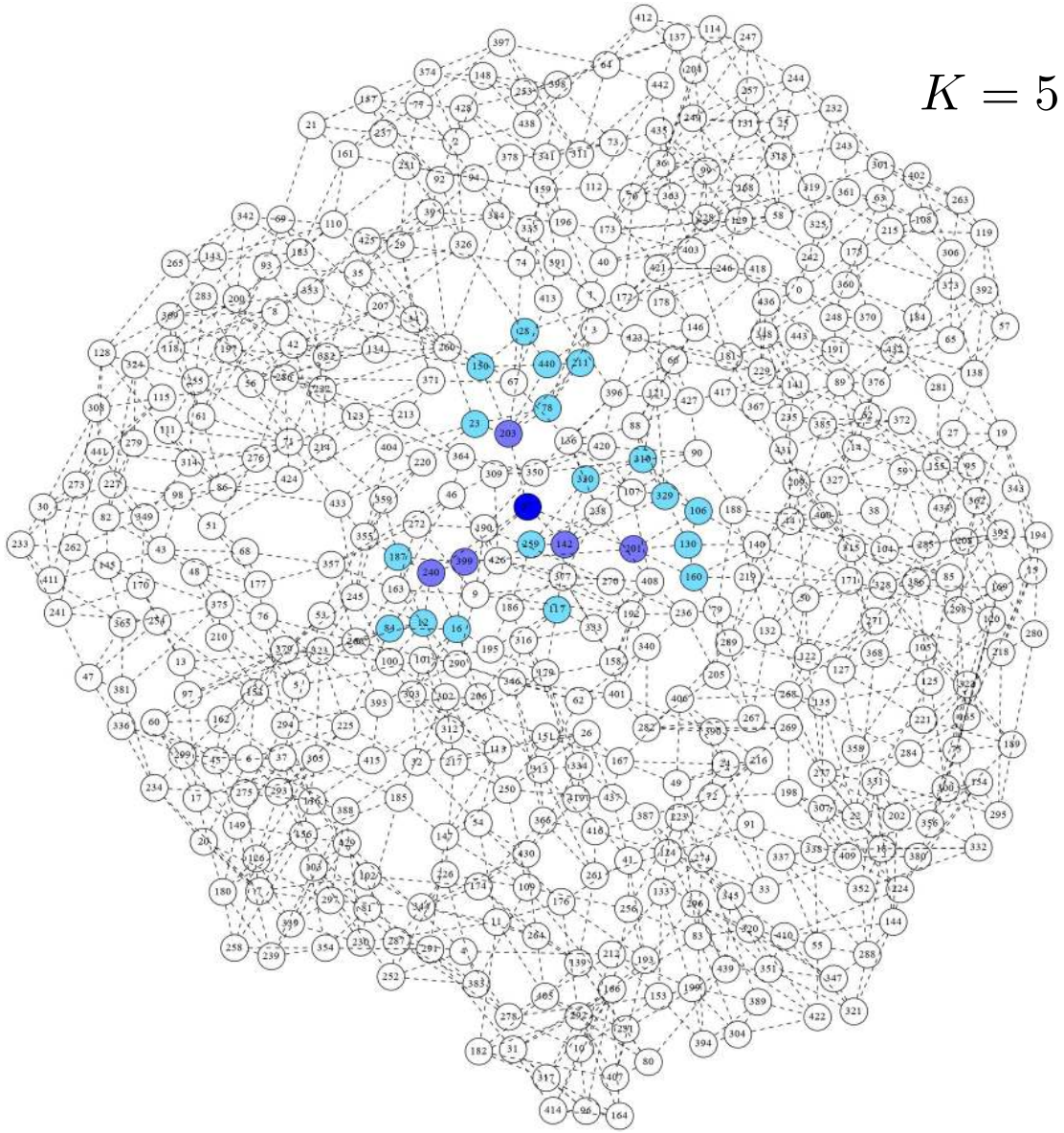




$\mathcal{G}_{\text{geo}}, \alpha = 1.0, \beta = 0.7, \gamma = 0.5$
444 vertices
C++ implementation w/ backtracking

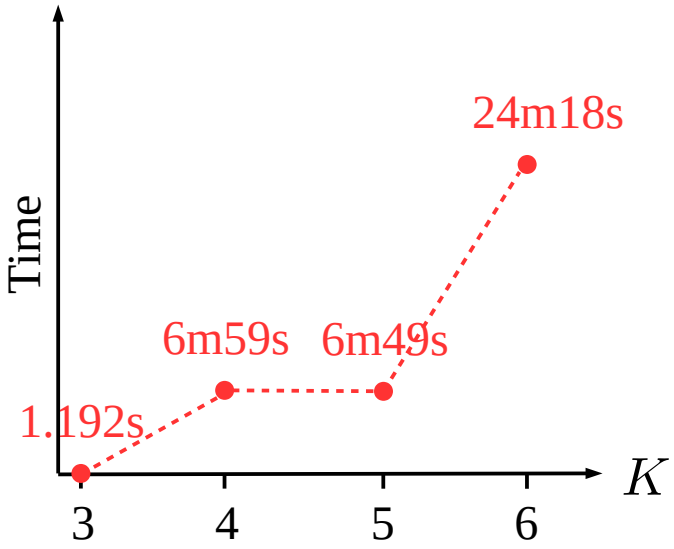
Supplementary — Time measurements — Function of K

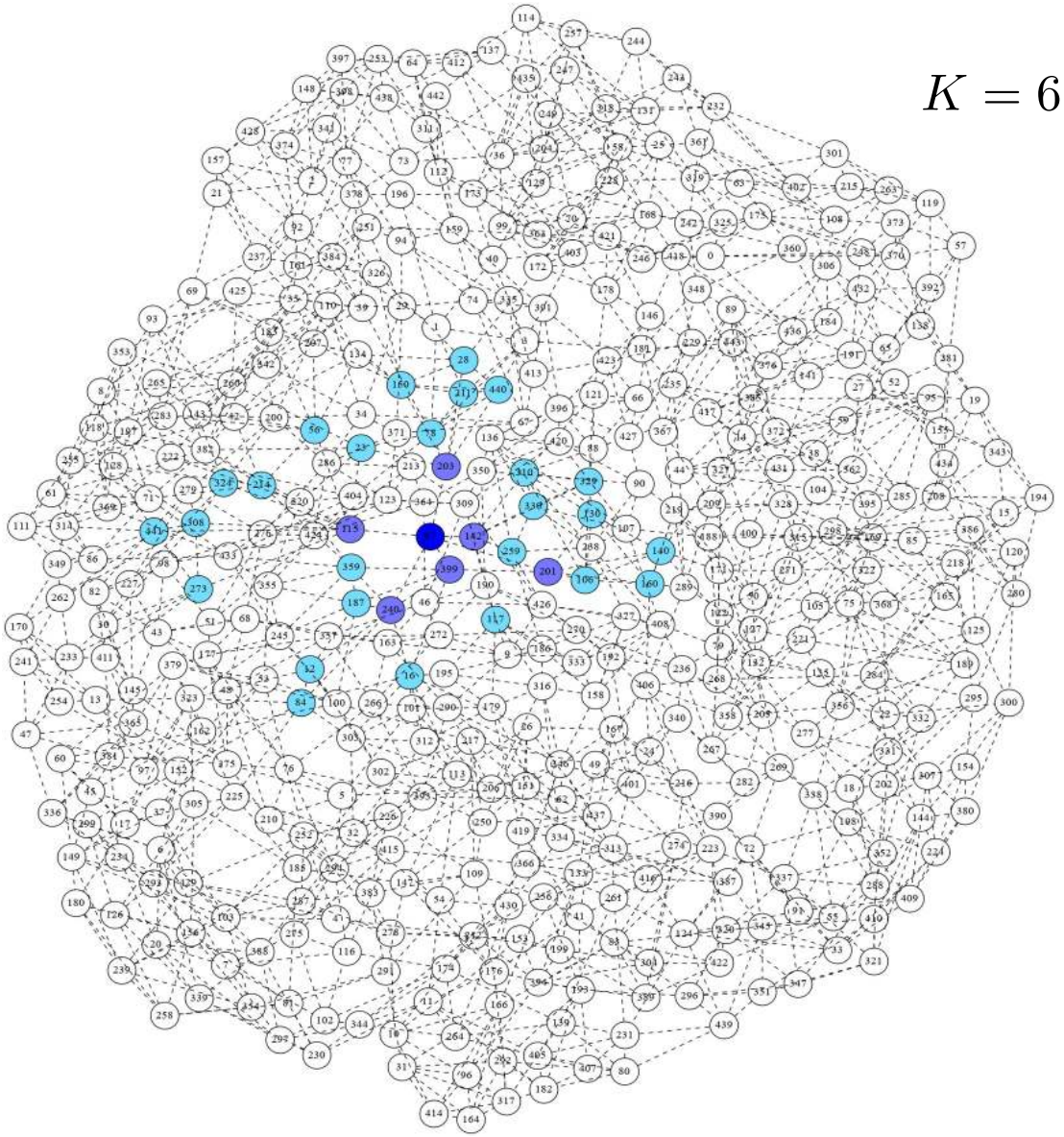




$\mathcal{G}_{\text{geo}}, \alpha = 1.0, \beta = 0.7, \gamma = 0.5$
444 vertices
C++ implementation w/ backtracking

Supplementary — Time measurements — Function of K

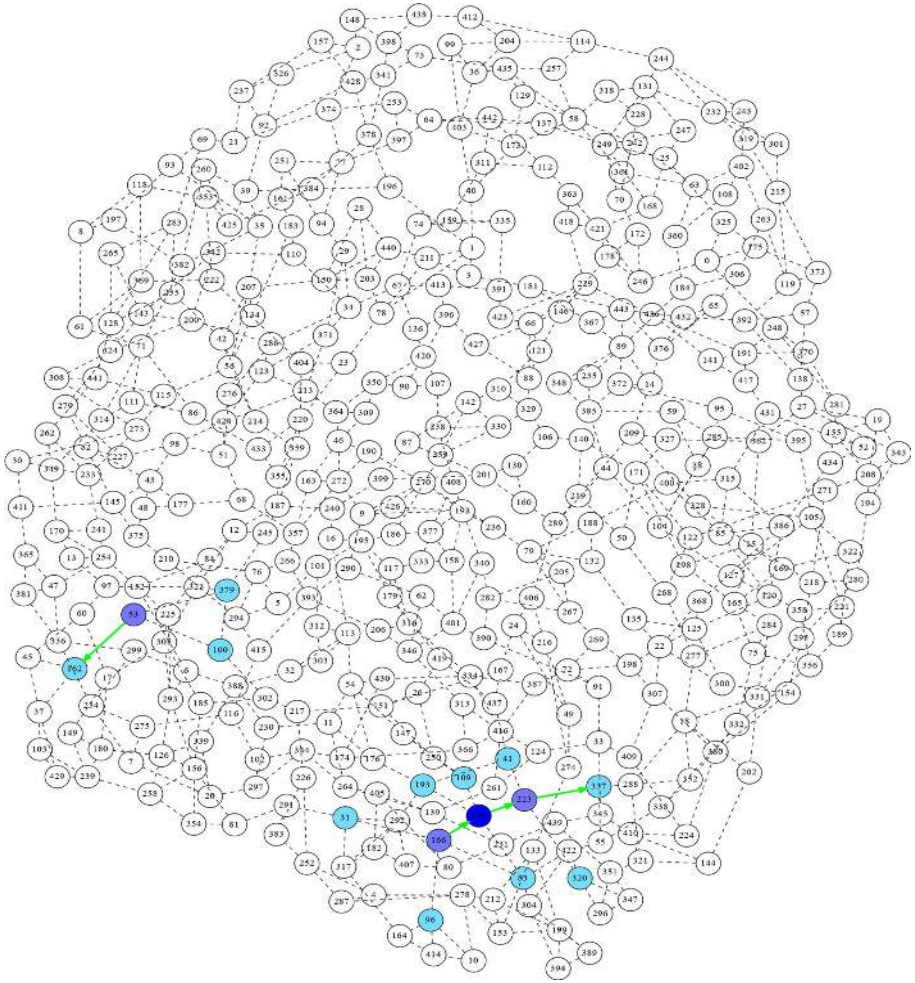


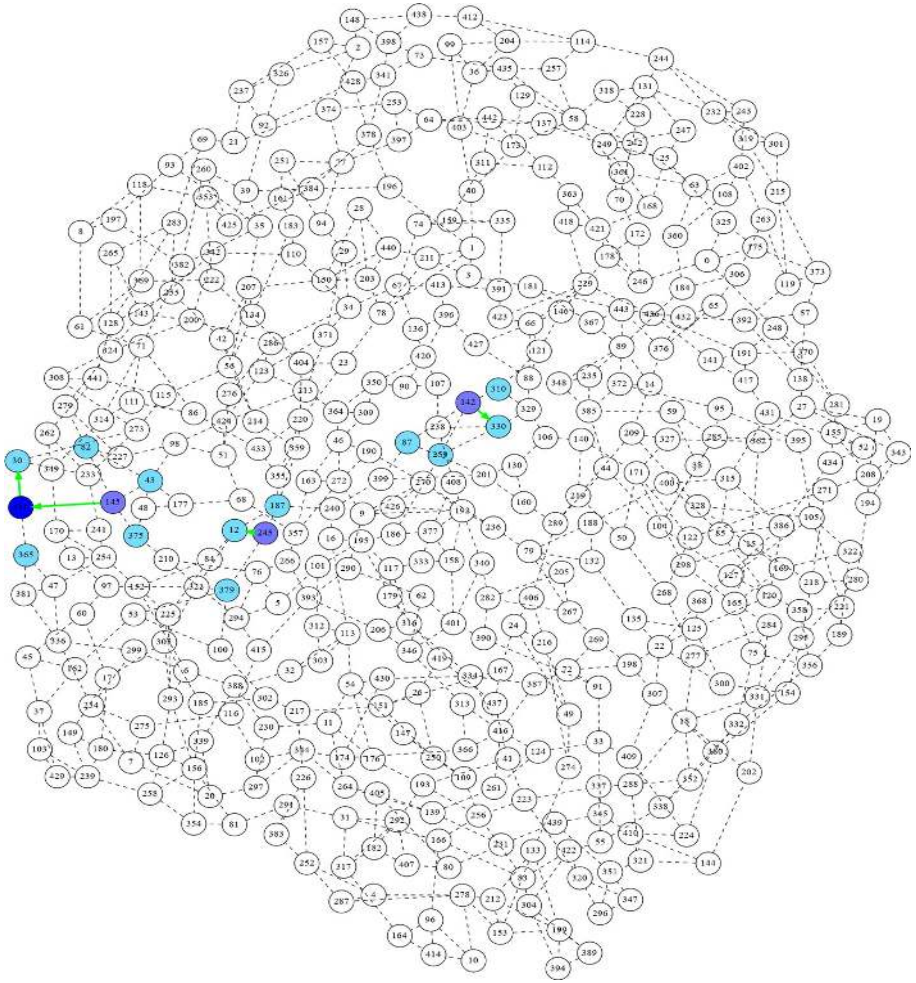


$\mathcal{G}_{\text{geo}}, \alpha = 1.0, \beta = 0.7, \gamma = 0.5$
444 vertices
C++ implementation w/ backtracking

Supplementary — Time measurements — Impact of parameters

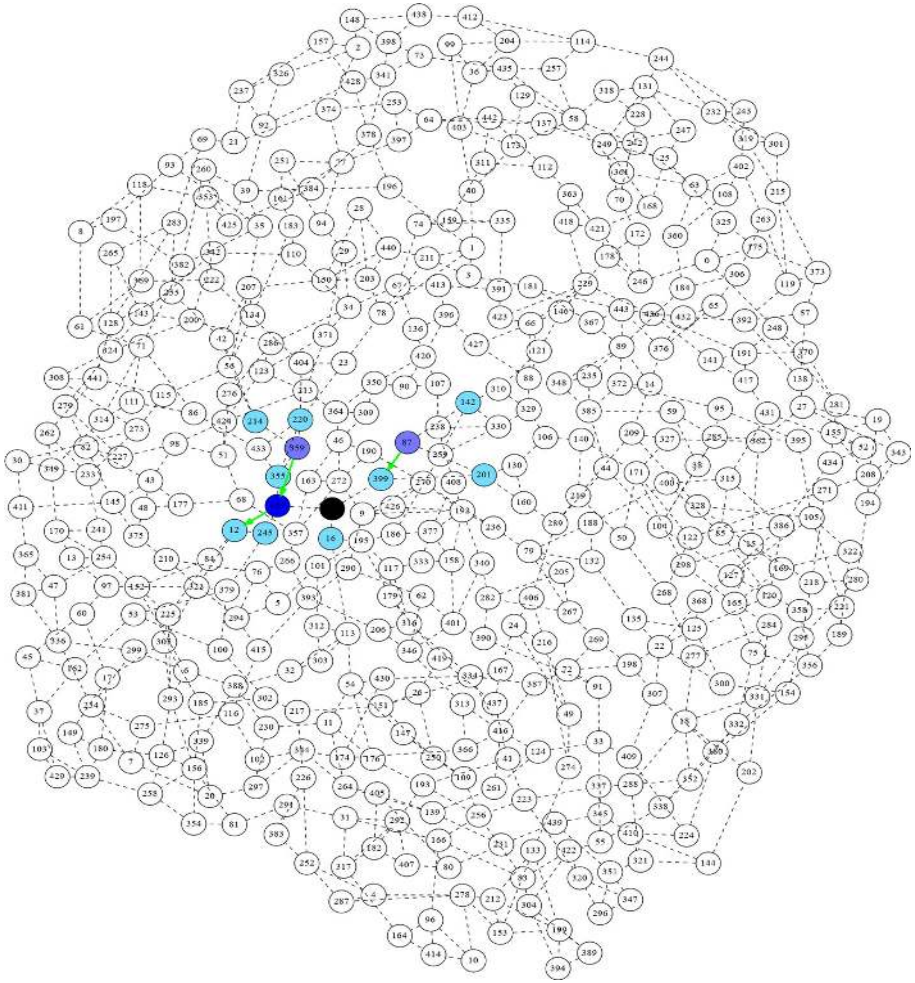
$$\mathcal{G}_{\text{geo}}, K = 3, \alpha = \infty, \beta = 0.7, \gamma = 0.5$$

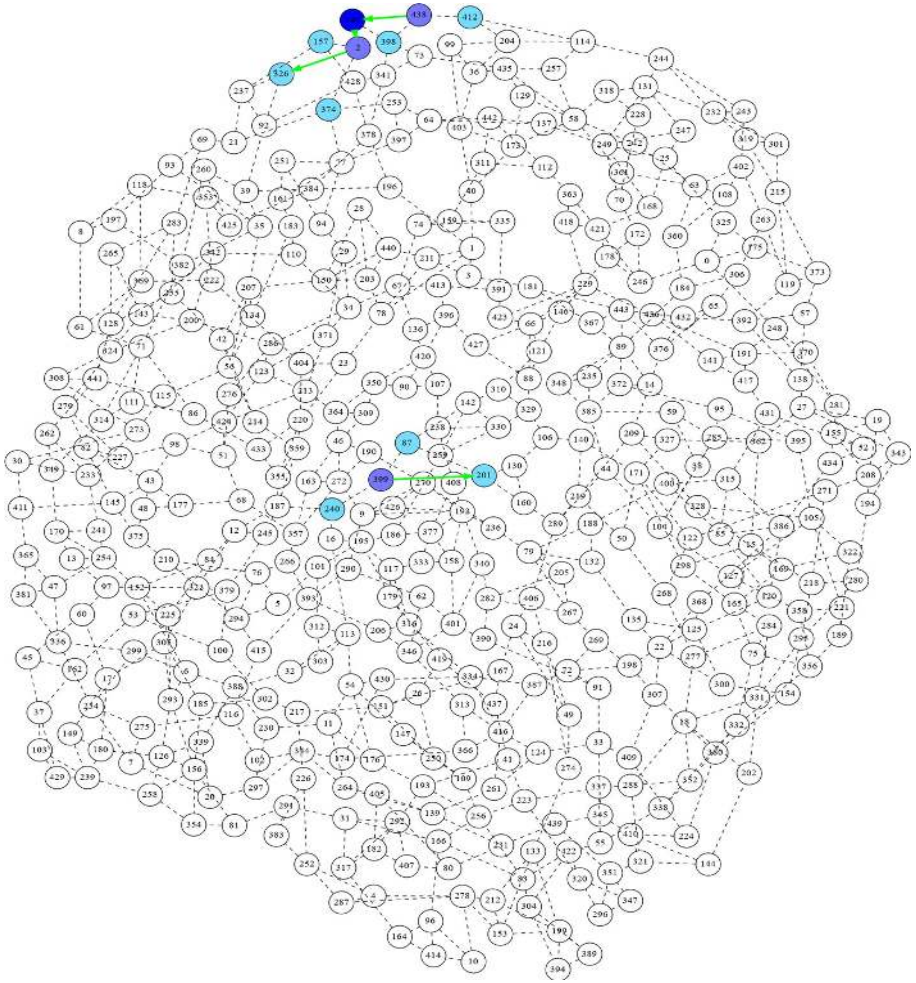




Supplementary — Time measurements — Impact of parameters

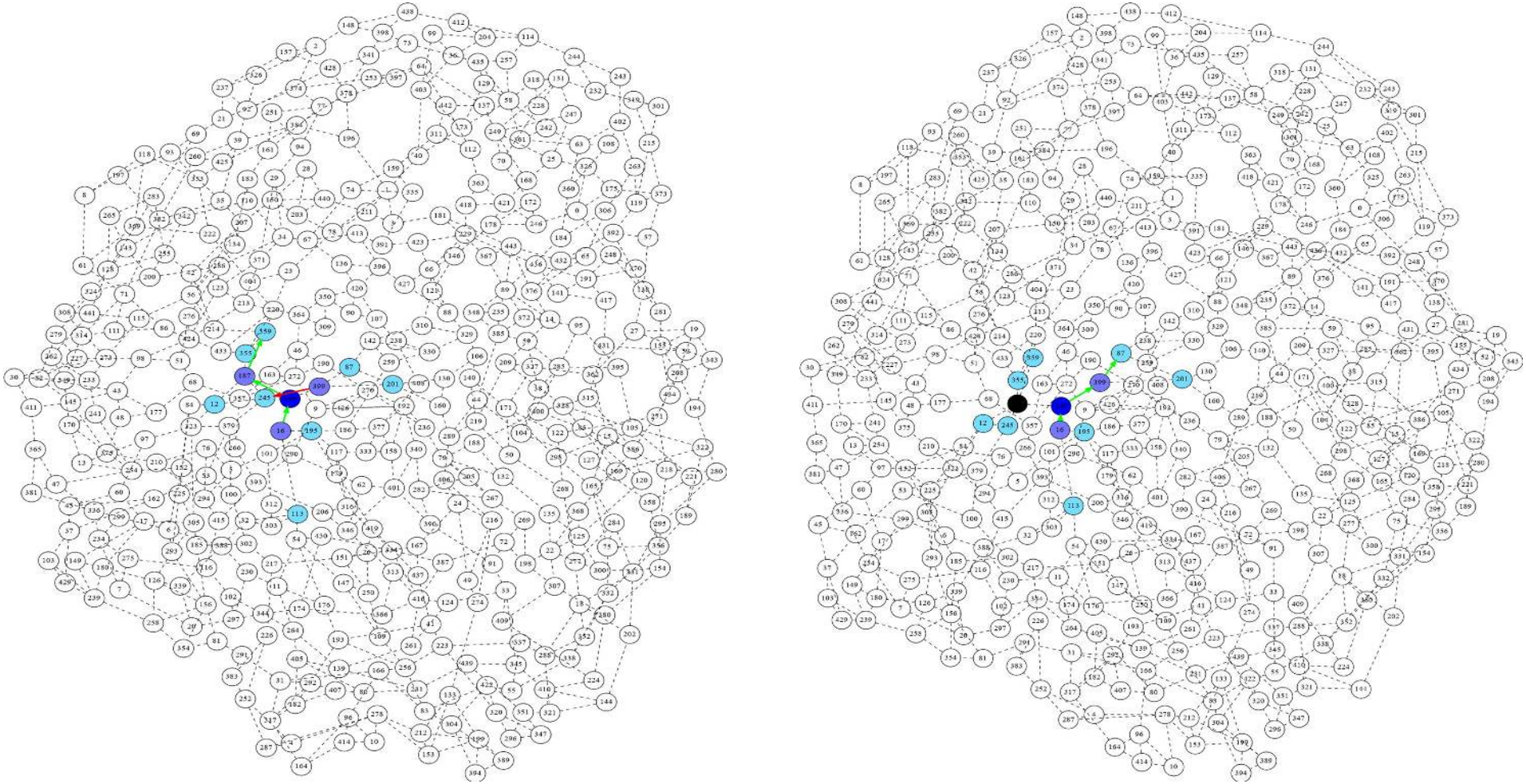
$$\mathcal{G}_{\text{geo}}, K = 3, \alpha = 1, 0, \beta = \infty, \gamma = 0.5$$





Supplementary — Time measurements — Impact of parameters

$$\mathcal{G}_{\text{geo}}, K = 3, \alpha = 1,0, \beta = 0,7, \gamma = \infty$$

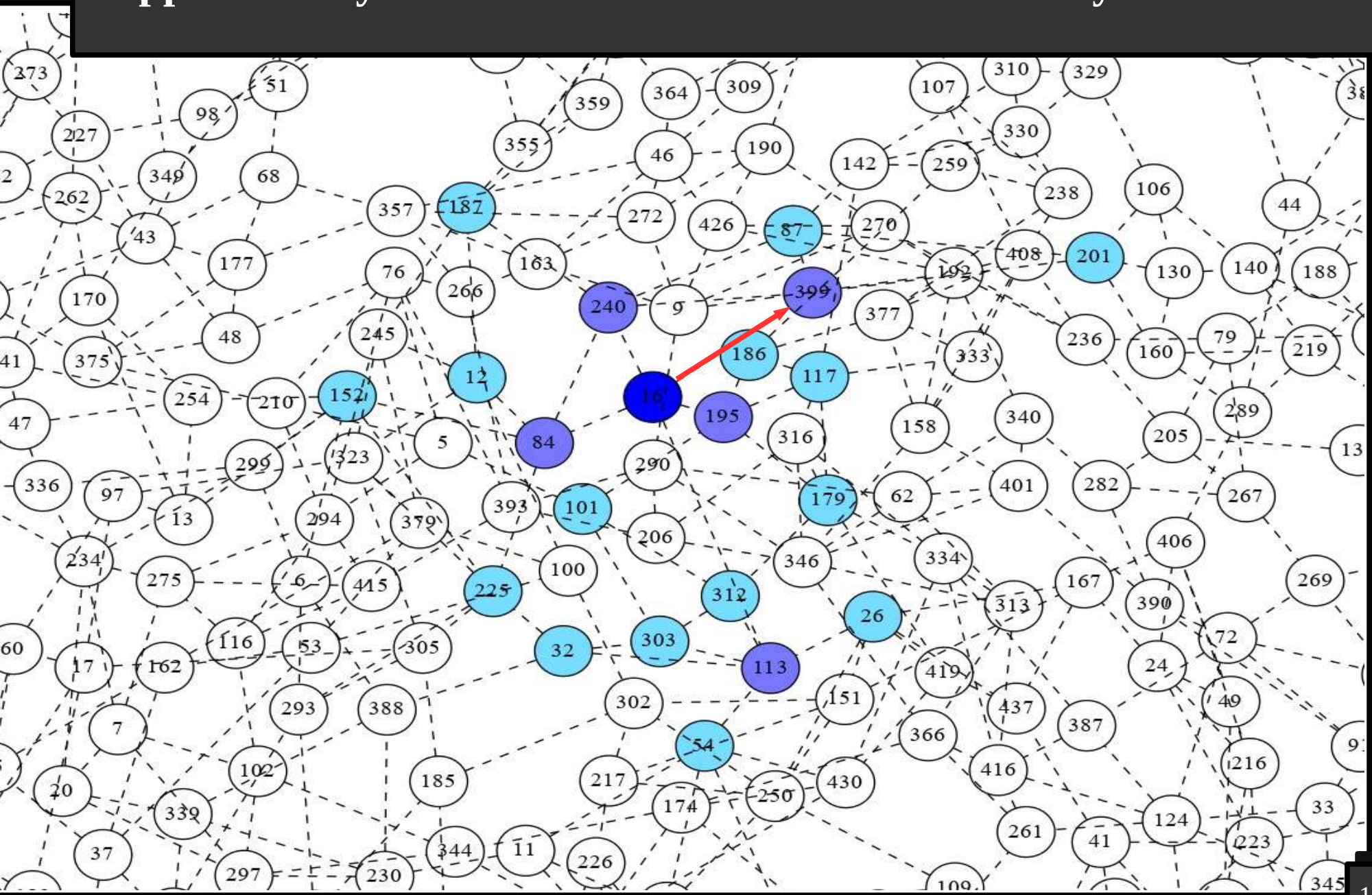


Outline

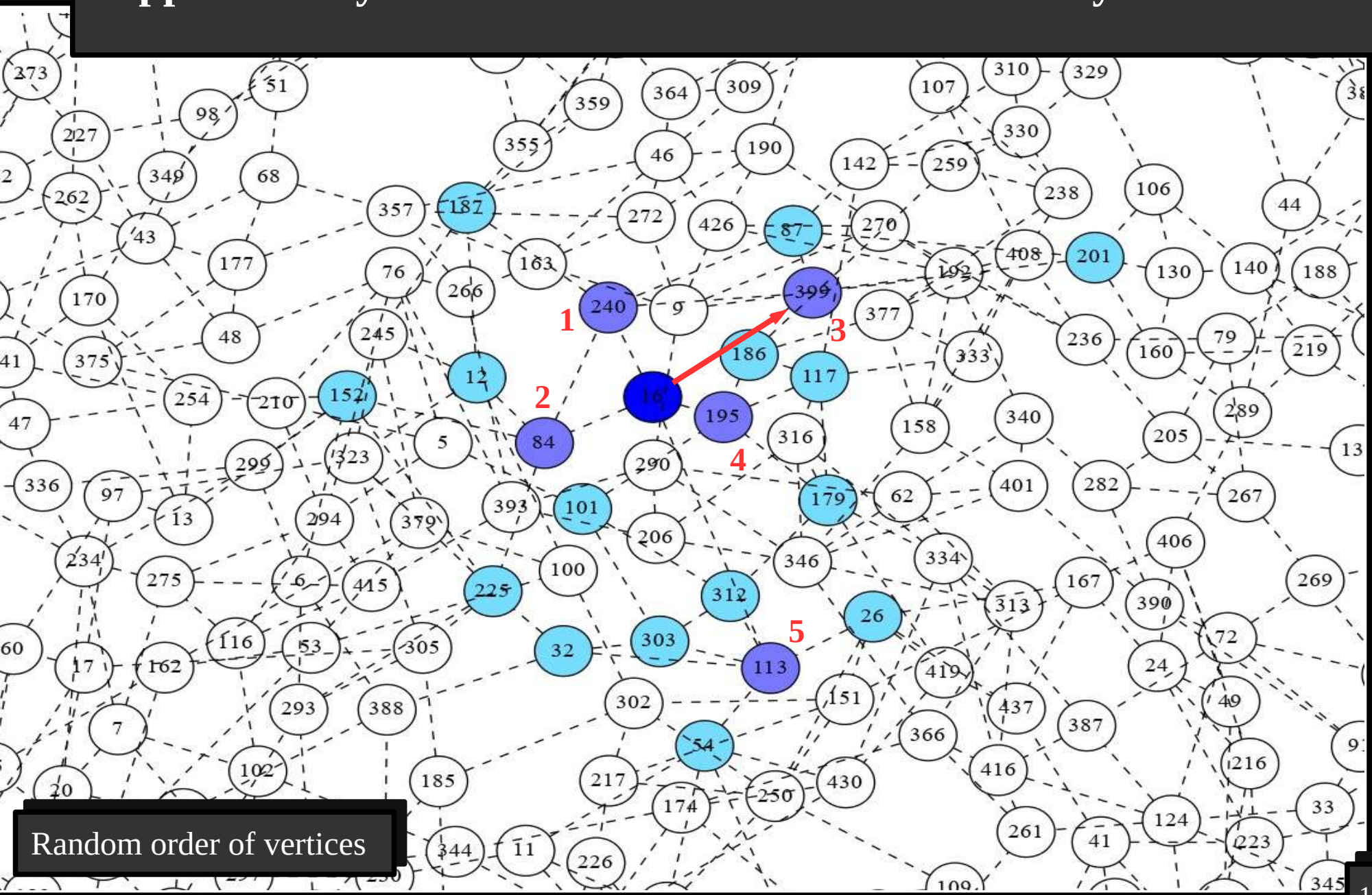
Supplementary slides

— Stochasticity to find translations —

Supplementary — Stochastic translations — Greedy w/ random order

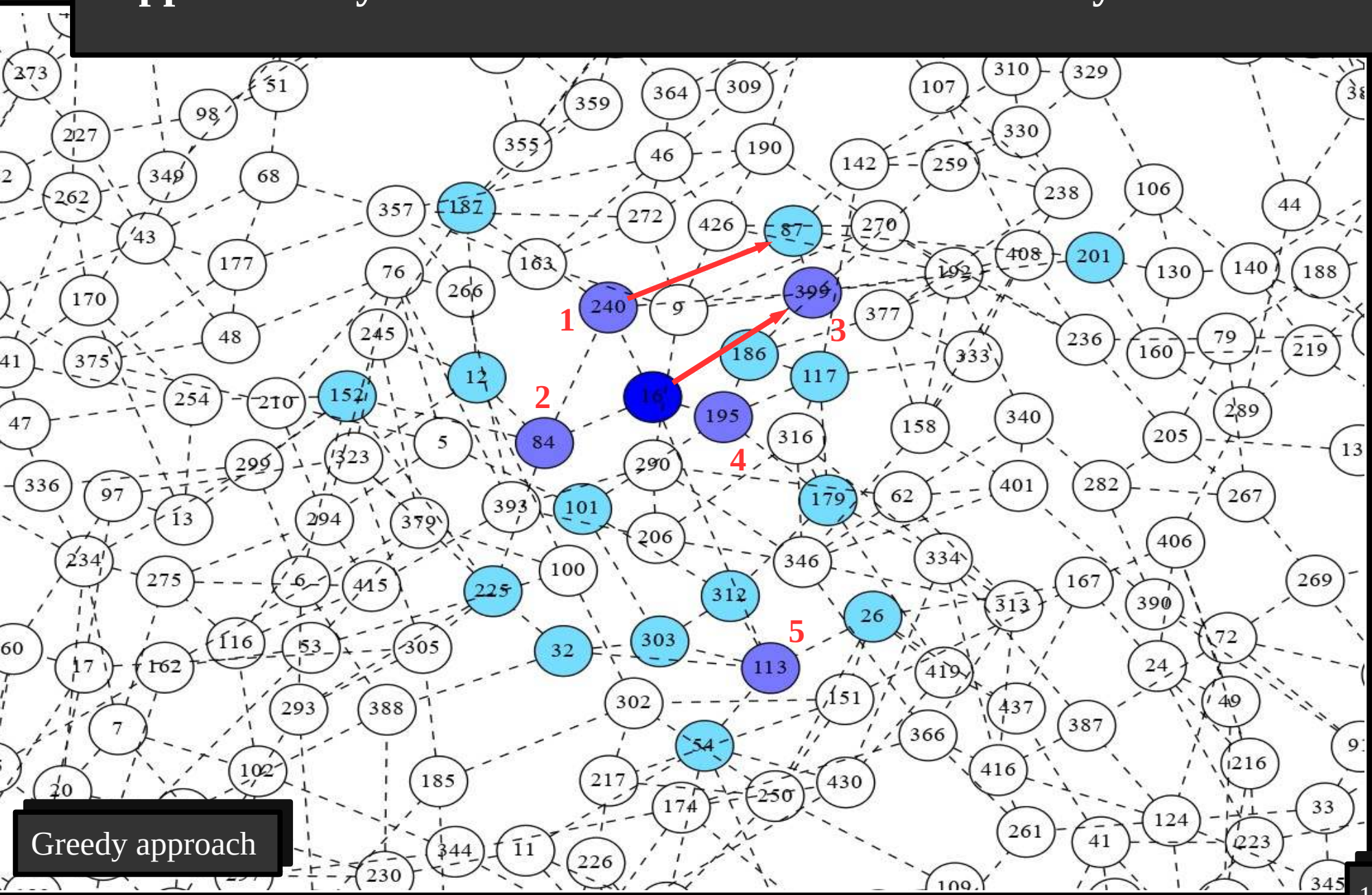


Supplementary — Stochastic translations — Greedy w/ random order



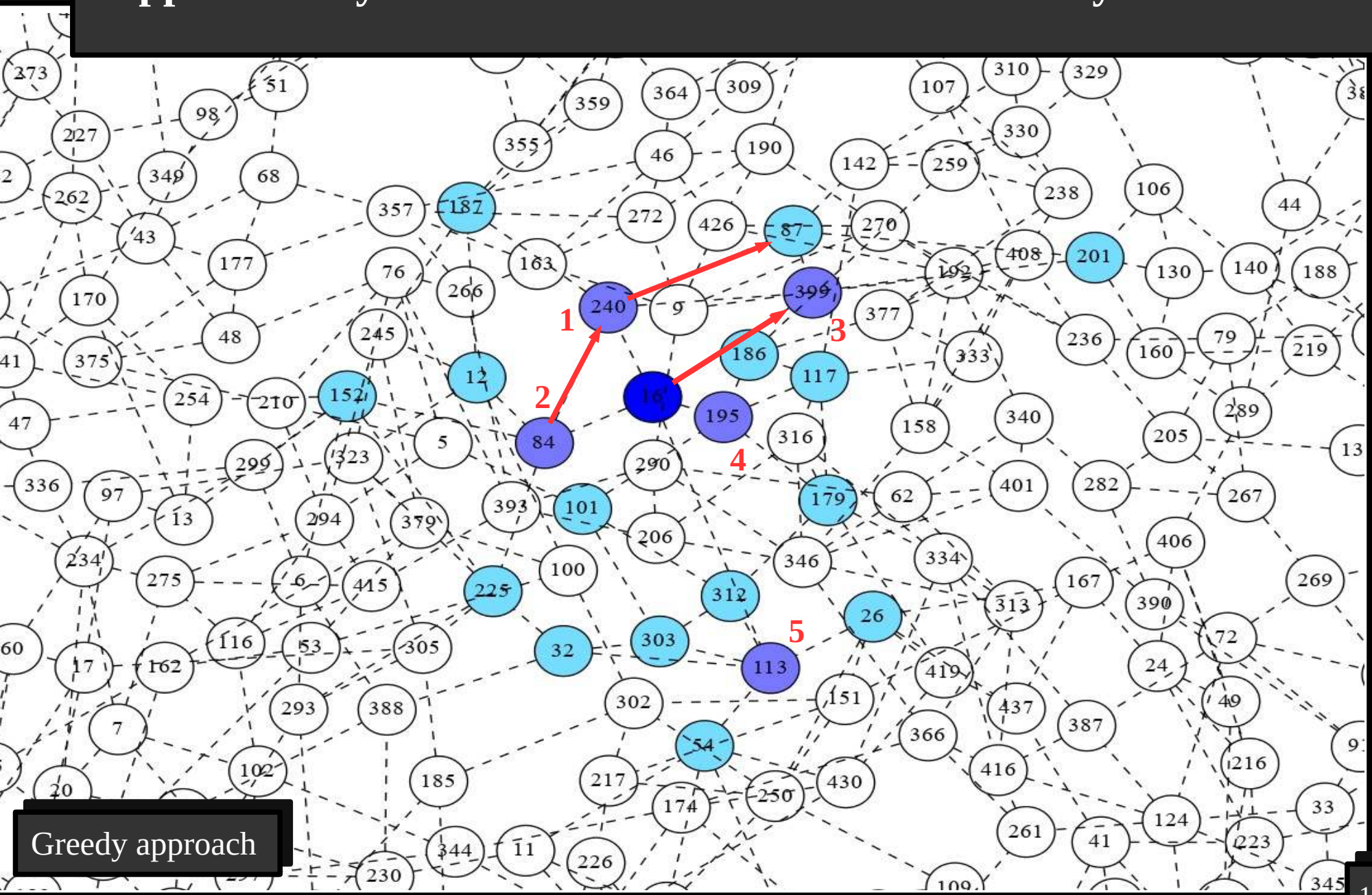
Random order of vertices

Supplementary — Stochastic translations — Greedy w/ random order



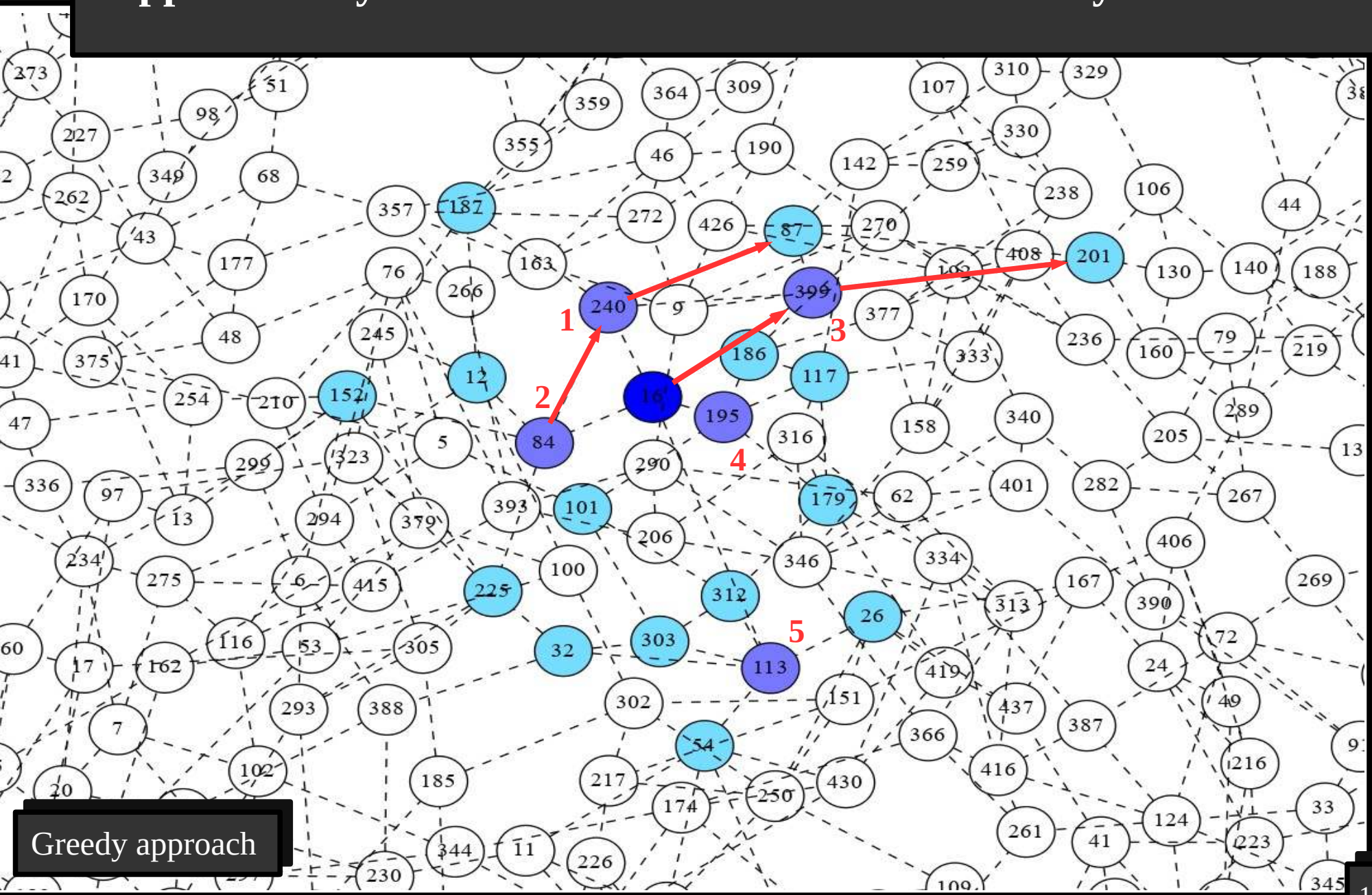
Greedy approach

Supplementary — Stochastic translations — Greedy w/ random order



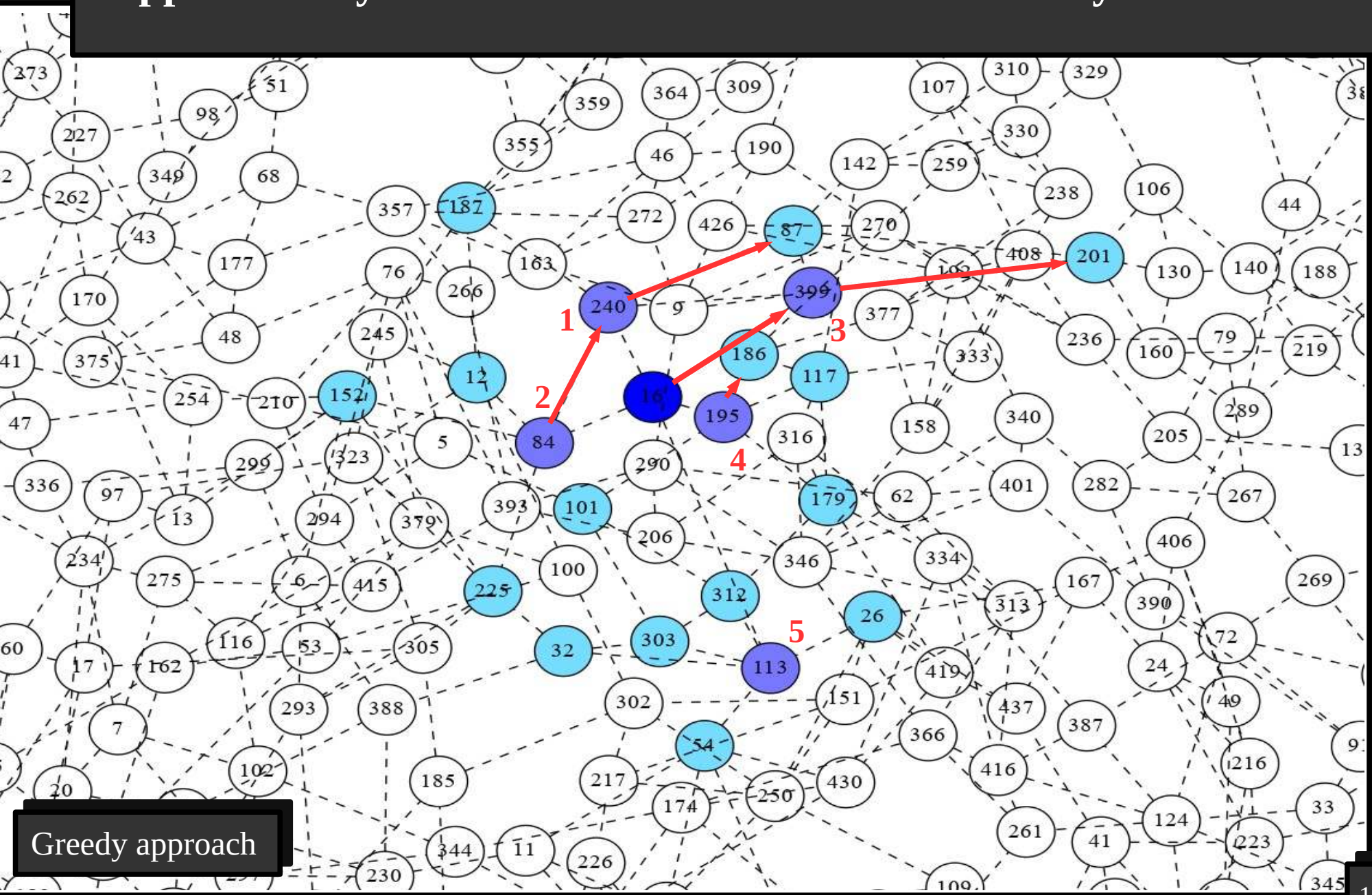
Greedy approach

Supplementary — Stochastic translations — Greedy w/ random order



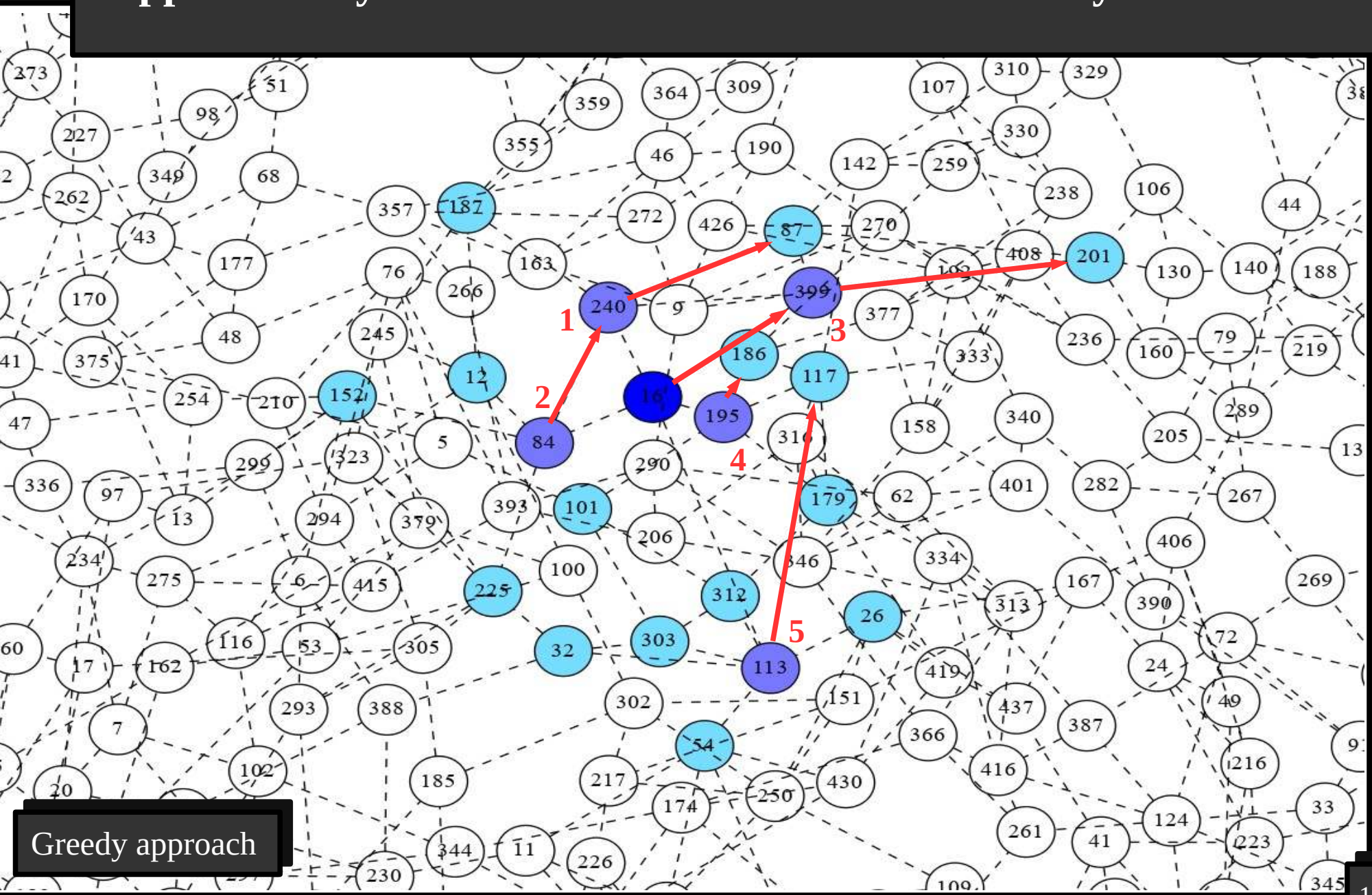
Greedy approach

Supplementary — Stochastic translations — Greedy w/ random order



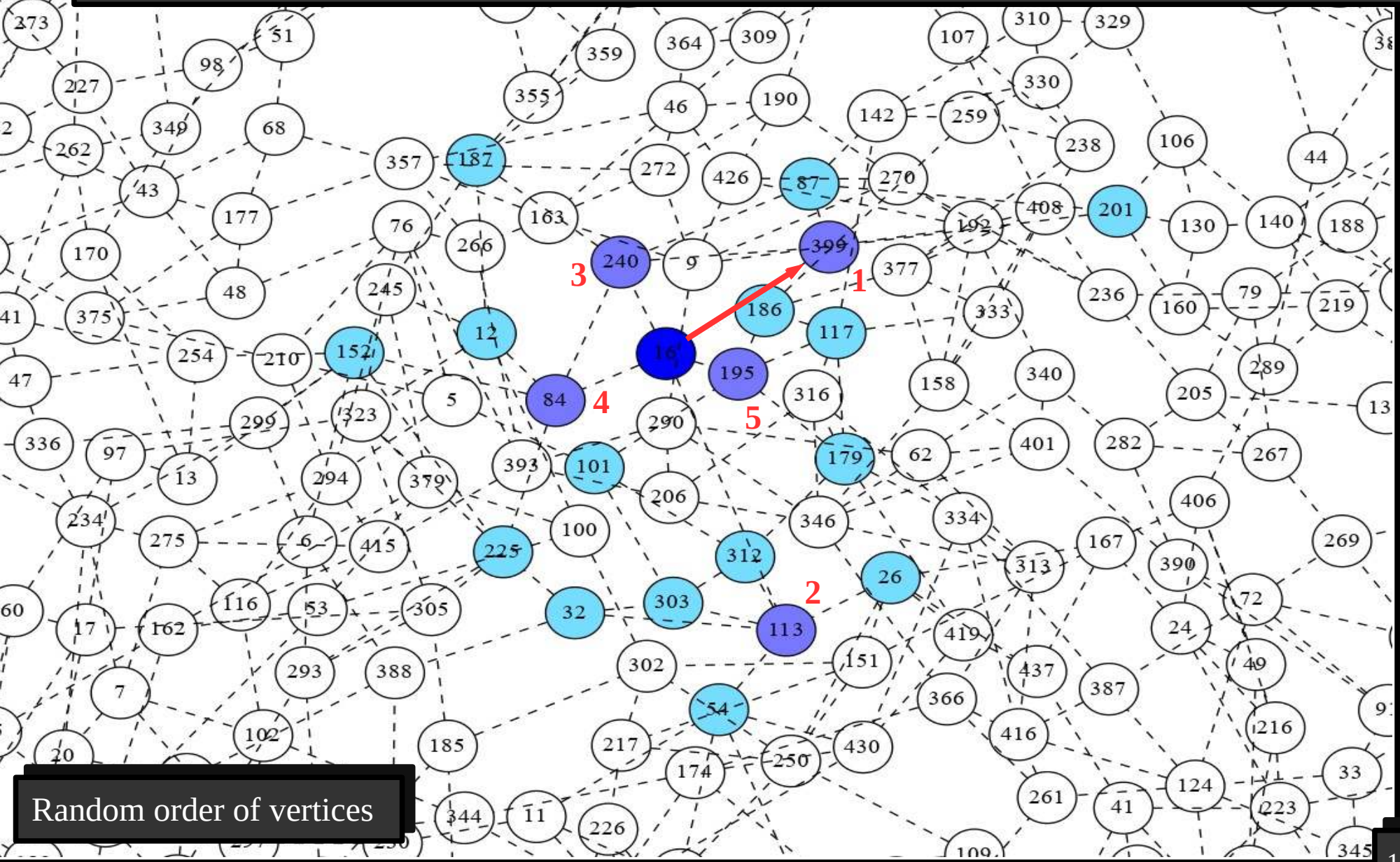
Greedy approach

Supplementary — Stochastic translations — Greedy w/ random order



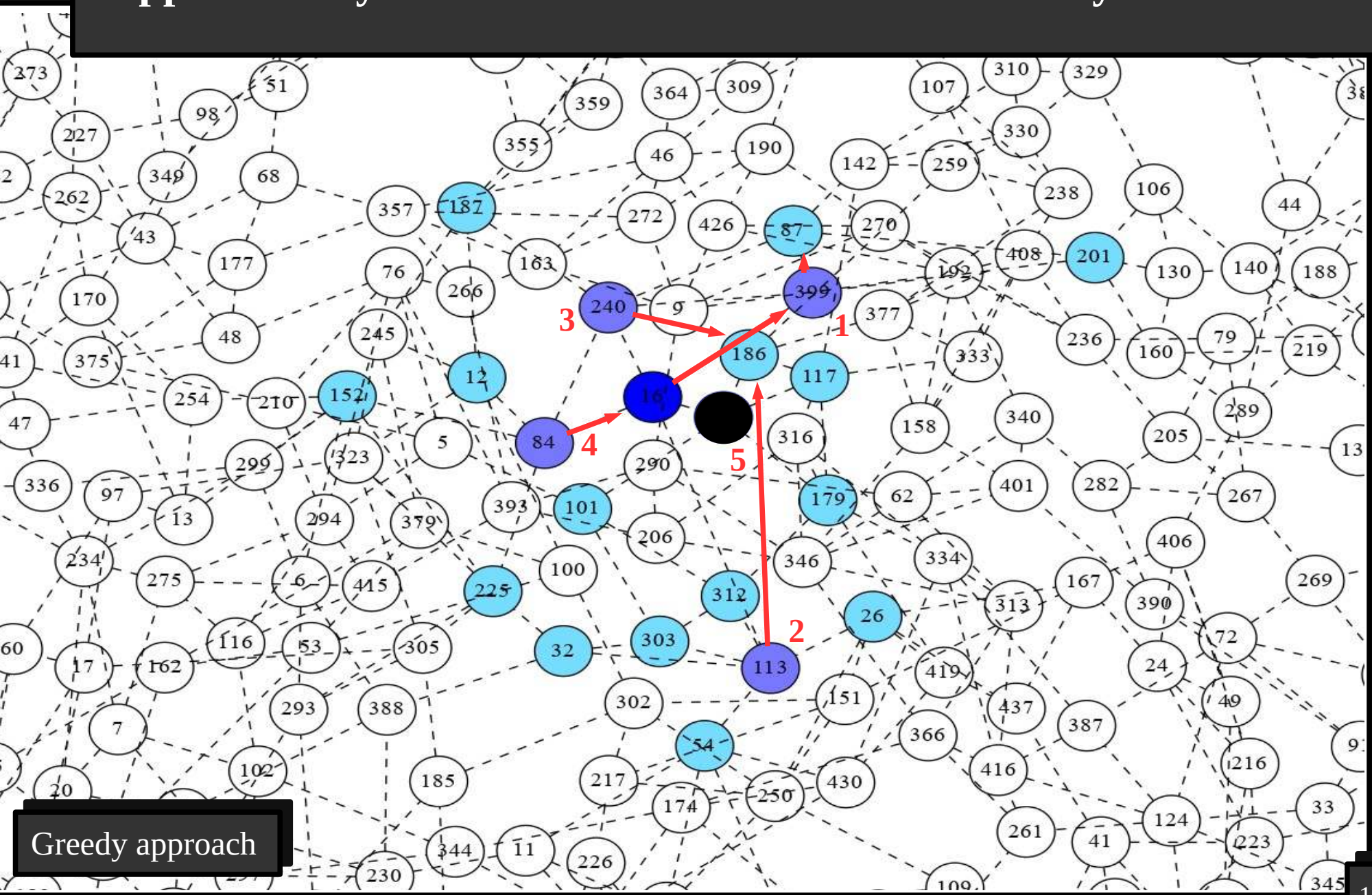
Greedy approach

Supplementary — Stochastic translations — Greedy w/ random order



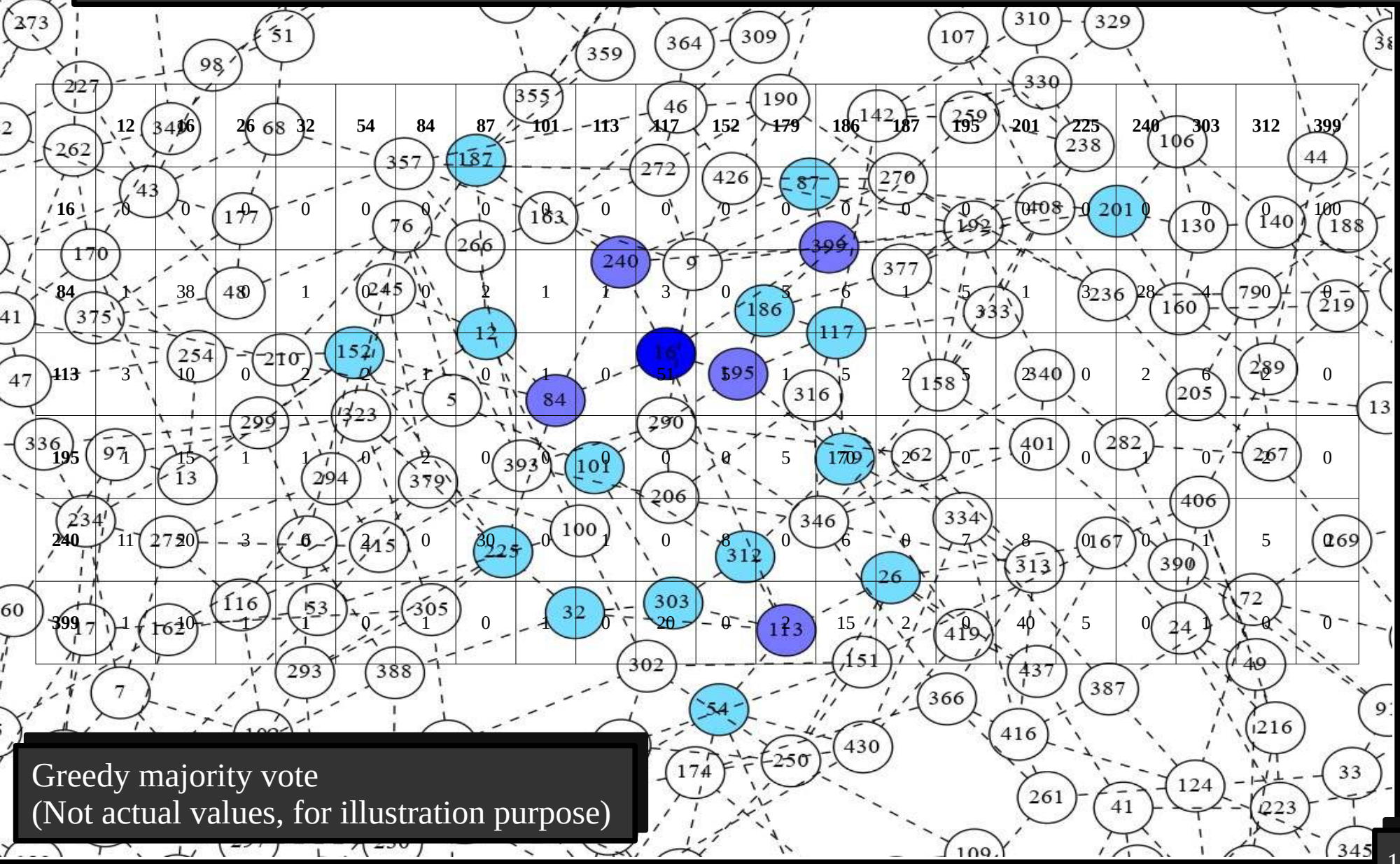
Random order of vertices

Supplementary — Stochastic translations — Greedy w/ random order



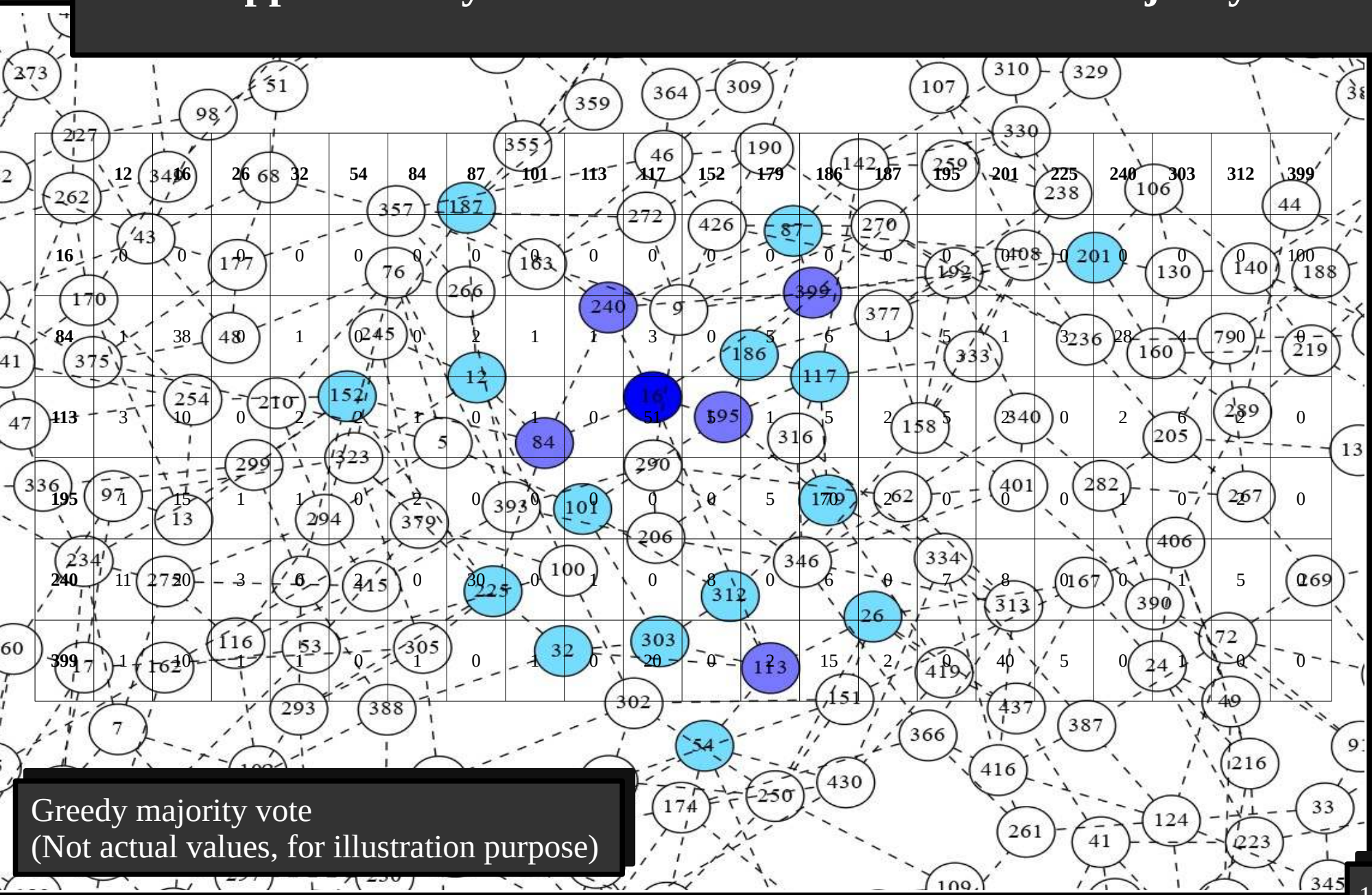
Greedy approach

Supplementary — Stochastic translations — Majority vote



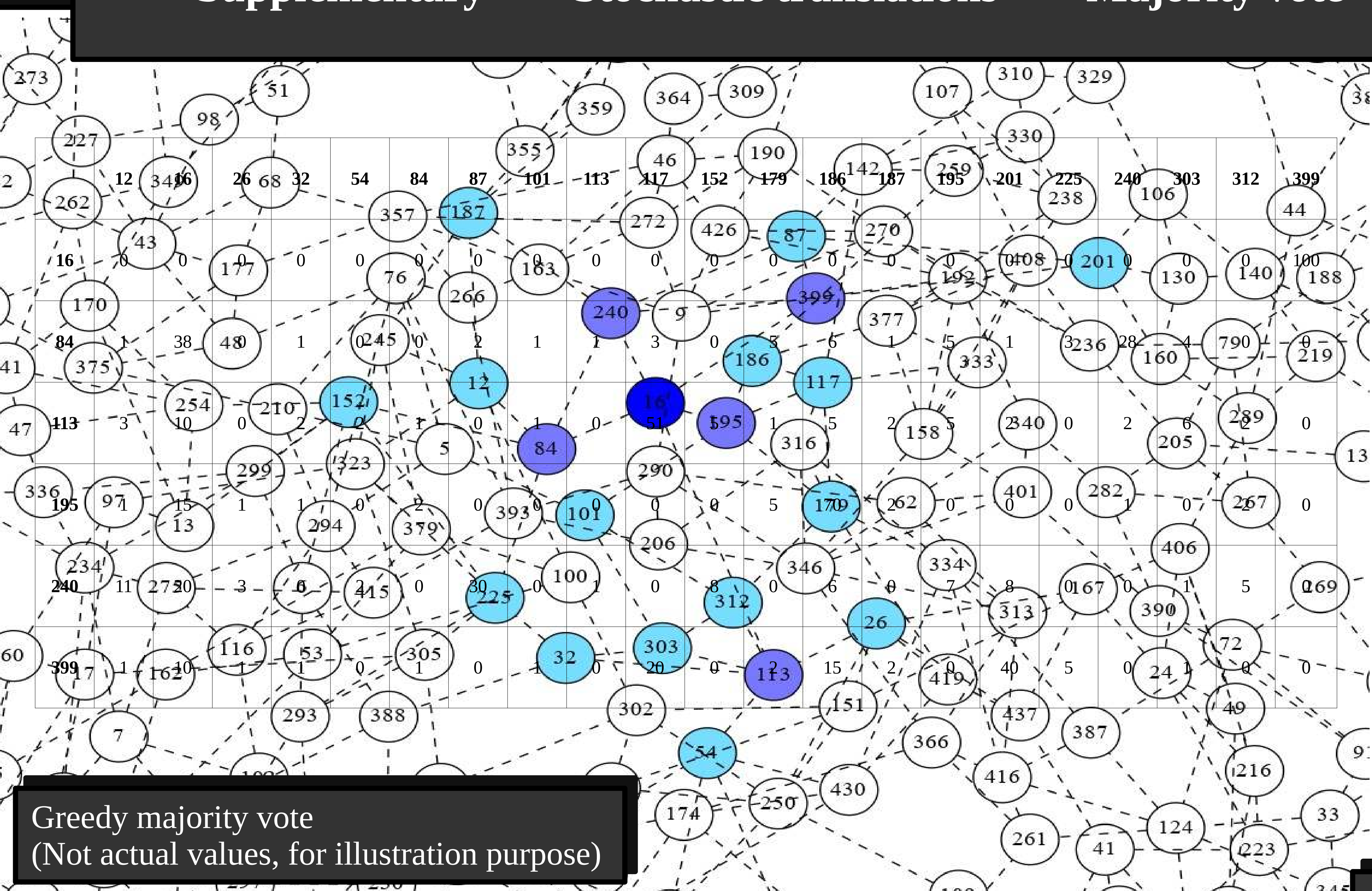
Greedy majority vote
(Not actual values, for illustration purpose)

Supplementary — Stochastic translations — Majority vote



Greedy majority vote
(Not actual values, for illustration purpose)

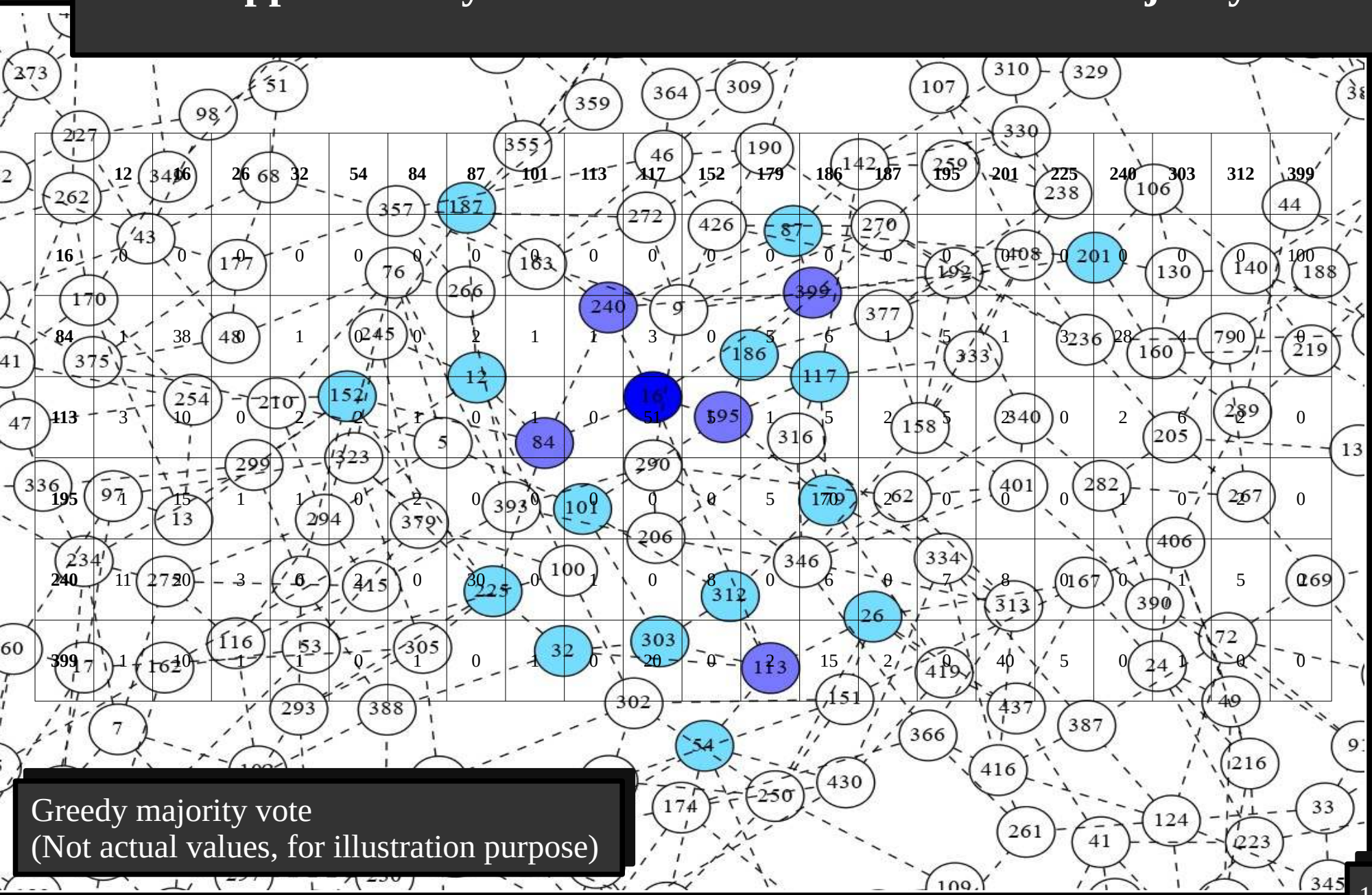
Supplementary — Stochastic translations — Majority vote



	12	16	26	32	54	84	87	101	113	117	152	179	186	187	195	201	225	240	303	312	399
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100
84	1	38	0	1	0	0	2	1	1	3	0	5	6	1	5	1	3	28	4	[illegible]	0
113	3	10	0	2	2	1	0	1	0	51	5	1	5	2	5	2	0	2	6	2	0
195	1	15	1	1	0	2	0	0	0	0	0	5	0	2	0	0	0	1	0	2	0
240	11	20	3	6	2	0	30	0	1	0	8	0	6	0	7	8	0	0	1	5	0
399	1	10	1	1	0	1	0	1	0	20	0	2	15	2	0	40	5	0	1	0	0

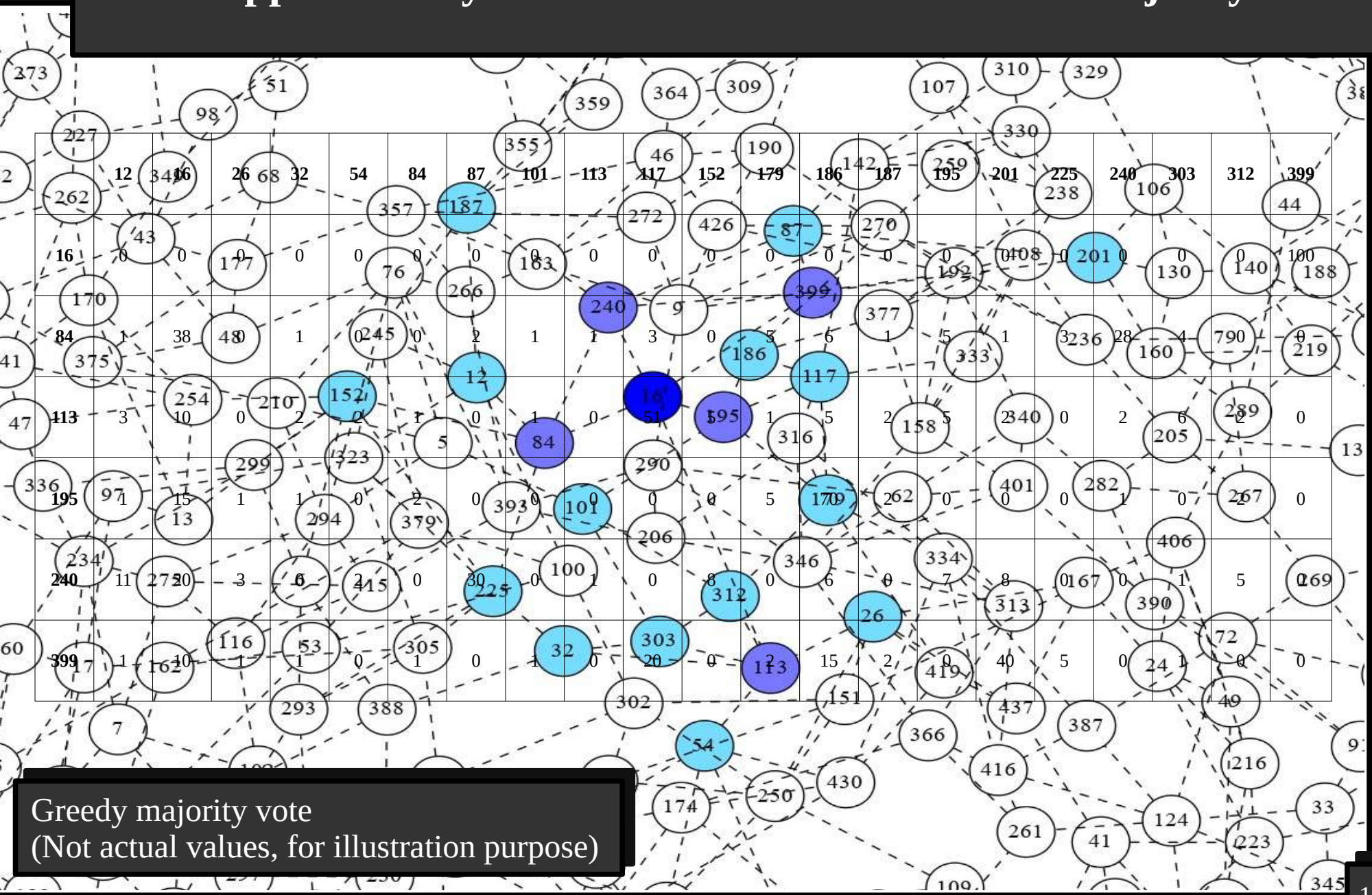
Greedy majority vote
(Not actual values, for illustration purpose)

Supplementary — Stochastic translations — Majority vote



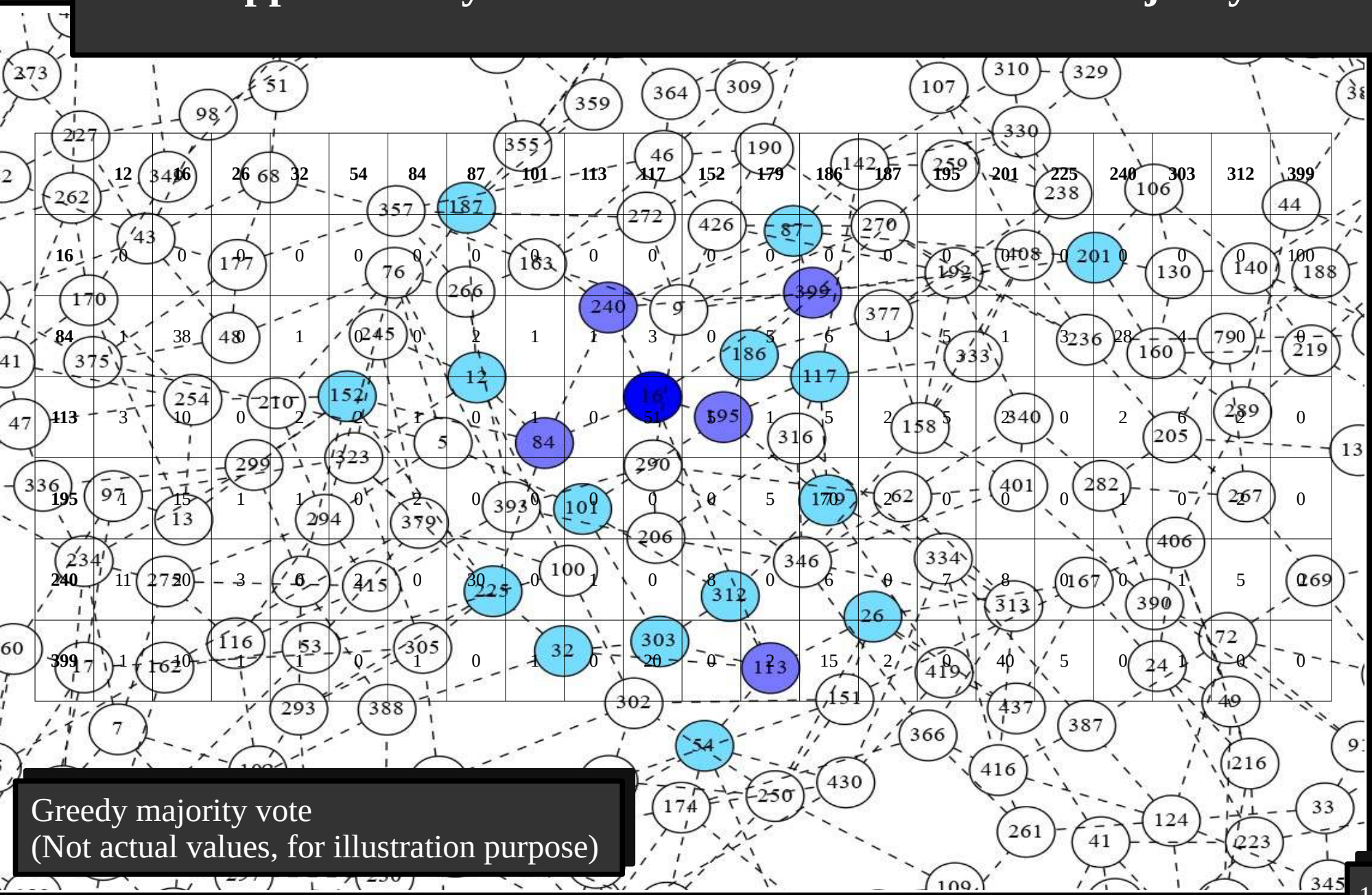
Greedy majority vote
(Not actual values, for illustration purpose)

Supplementary — Stochastic translations — Majority vote



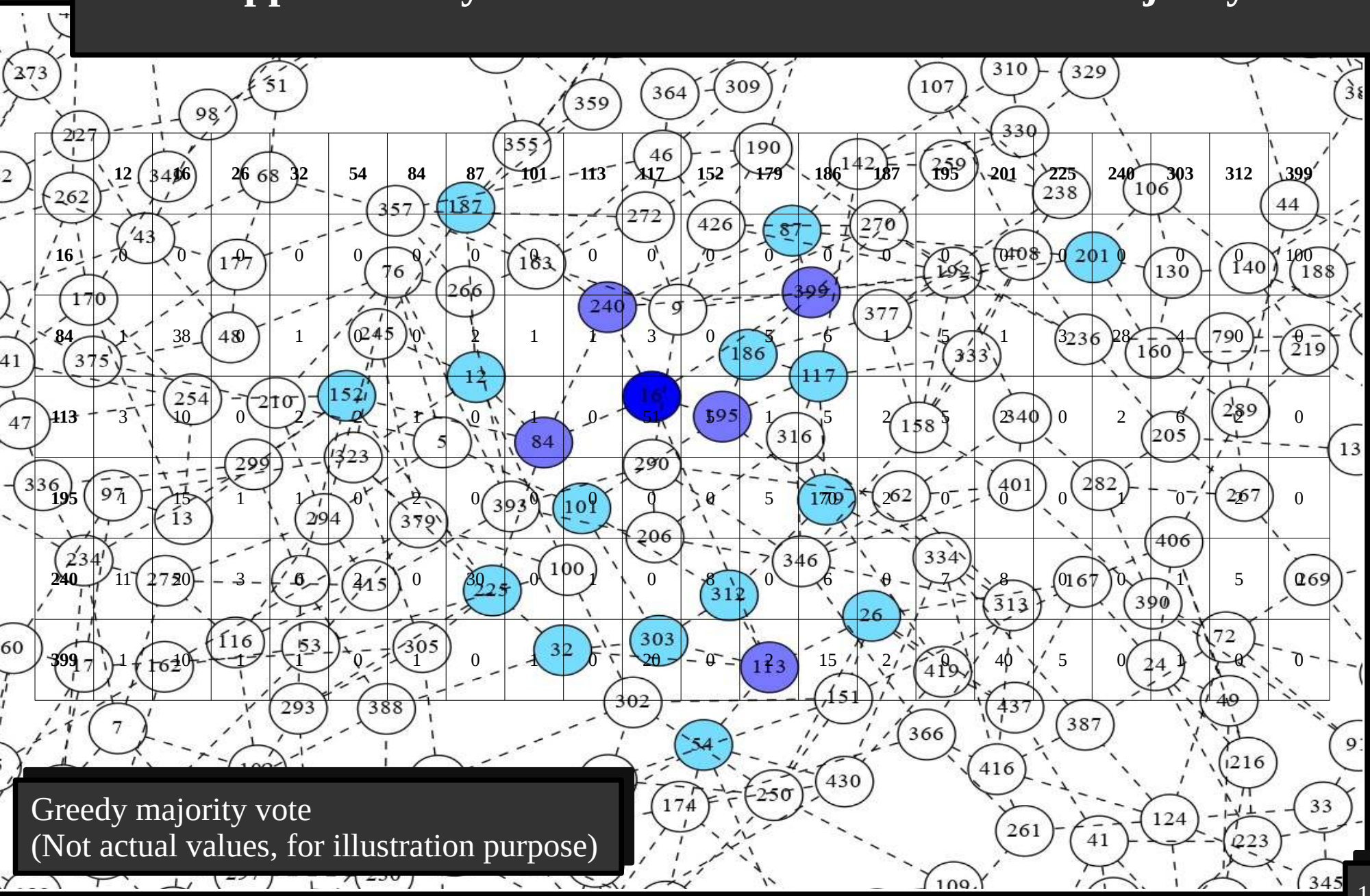
Greedy majority vote
(Not actual values, for illustration purpose)

Supplementary — Stochastic translations — Majority vote



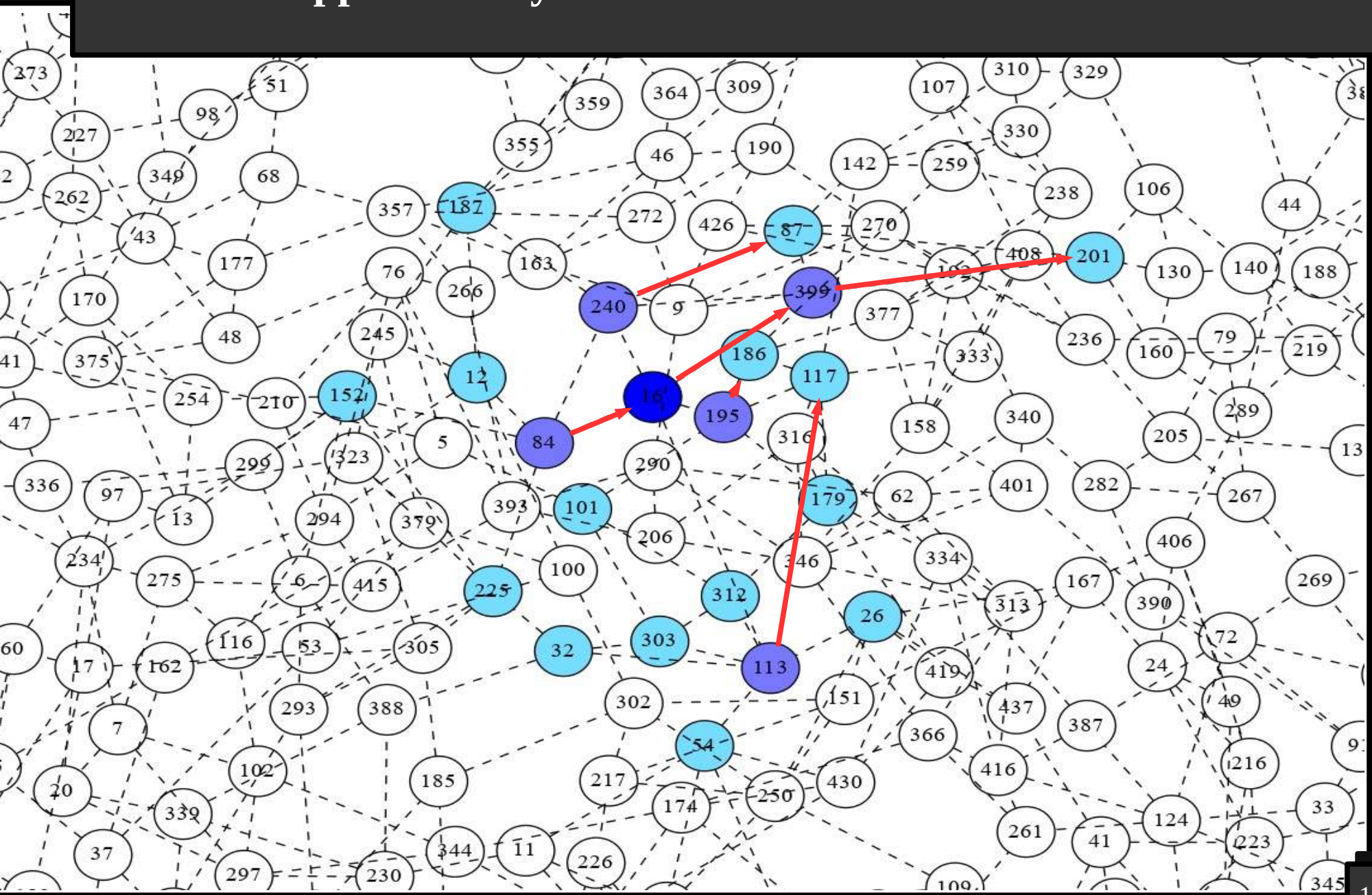
Greedy majority vote
(Not actual values, for illustration purpose)

Supplementary — Stochastic translations — Majority vote

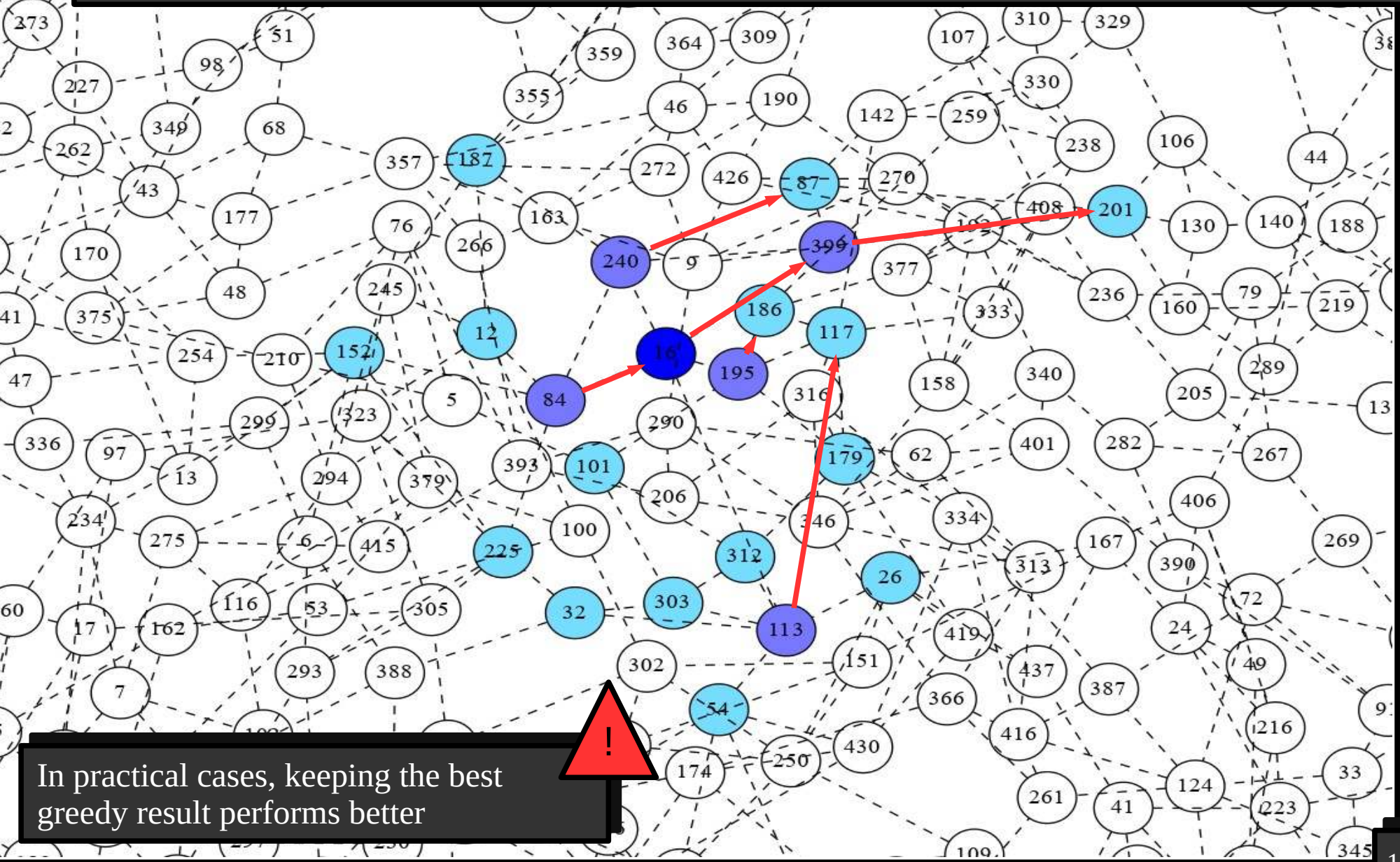


Greedy majority vote
(Not actual values, for illustration purpose)

Supplementary — Stochastic translations — Result



Supplementary — Stochastic translations — Result

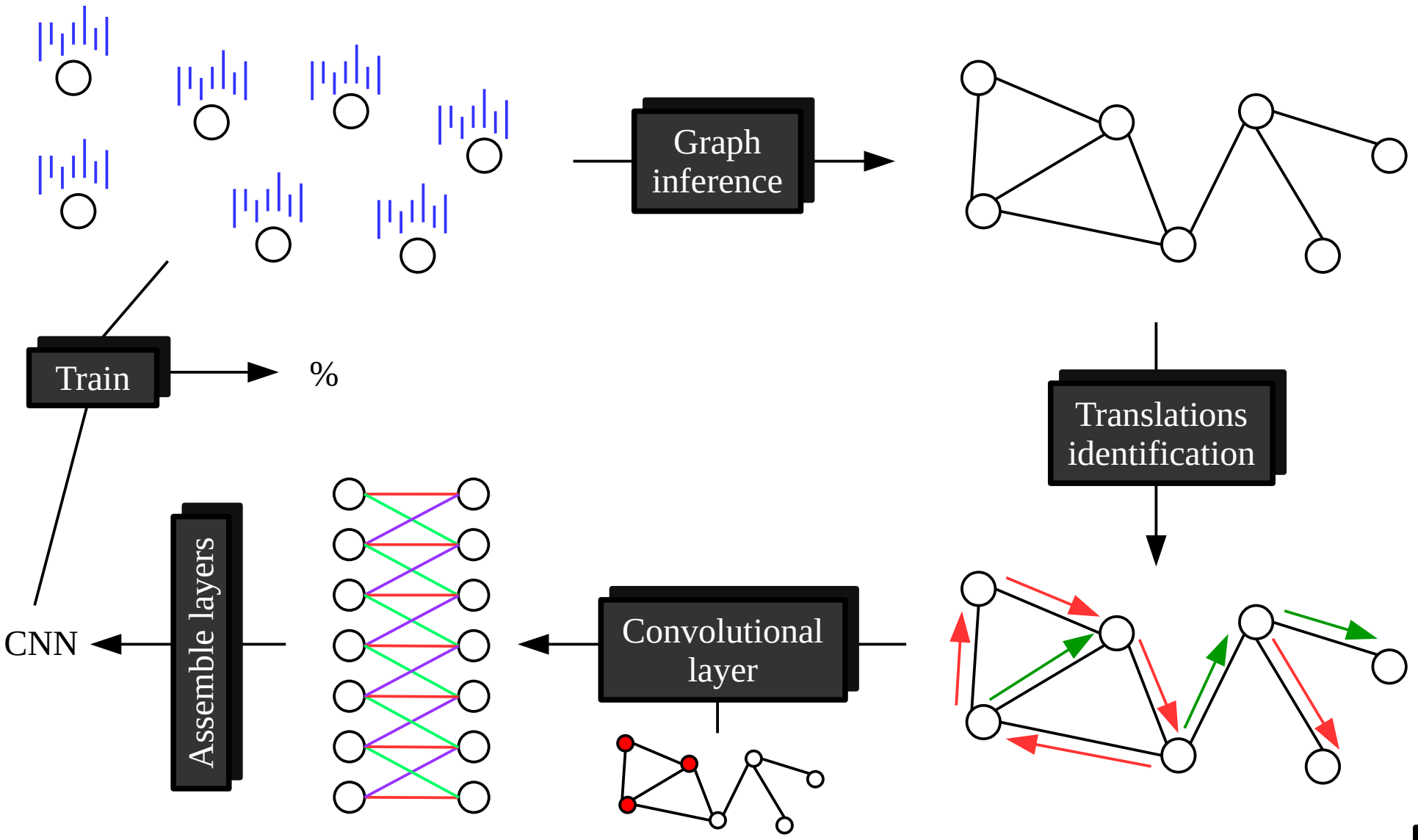


Outline

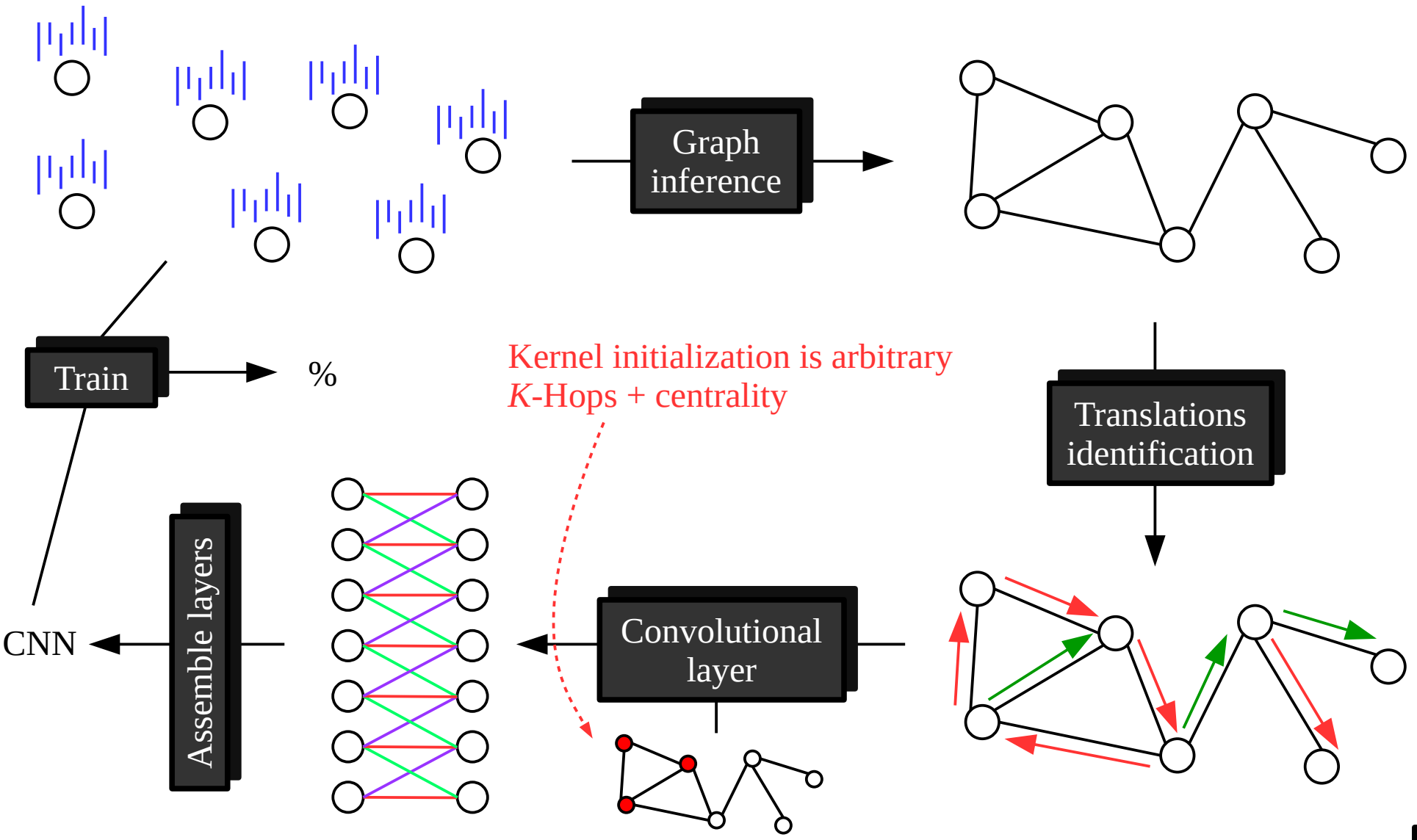
Supplementary slides

— Ideas on dictionary learning —

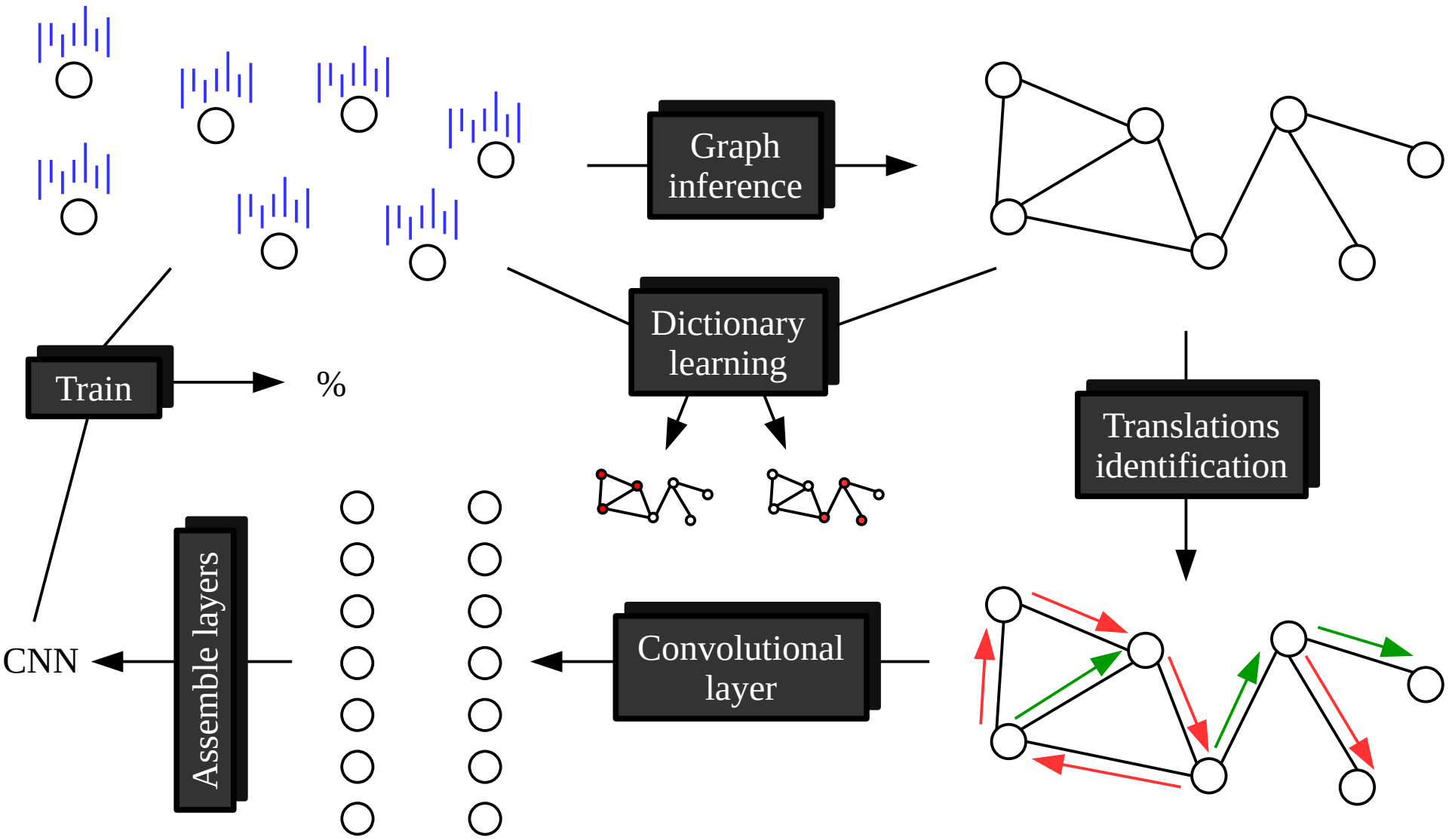
Supplementary — Dictionary learning — The idea



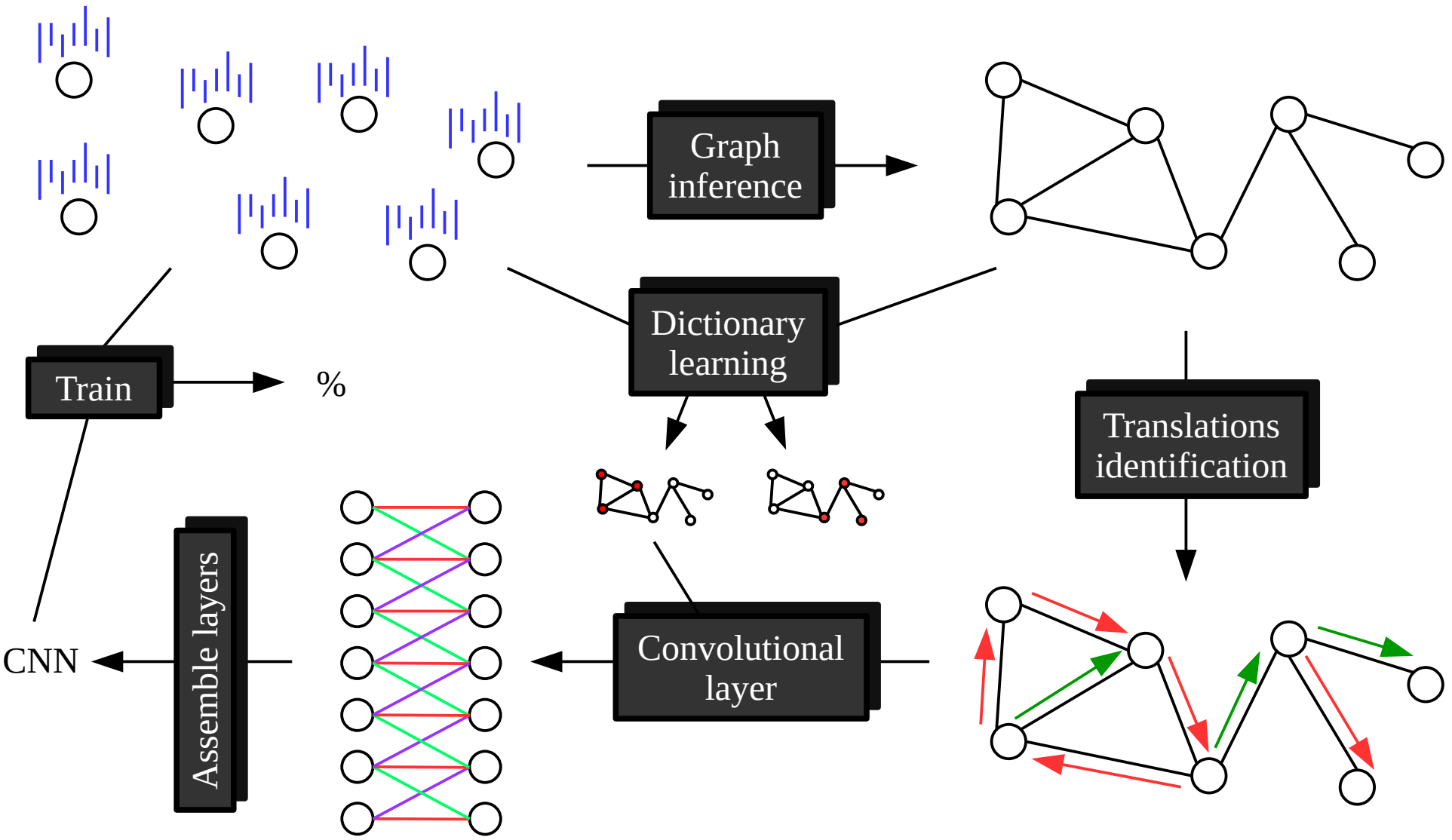
Supplementary — Dictionary learning — The idea



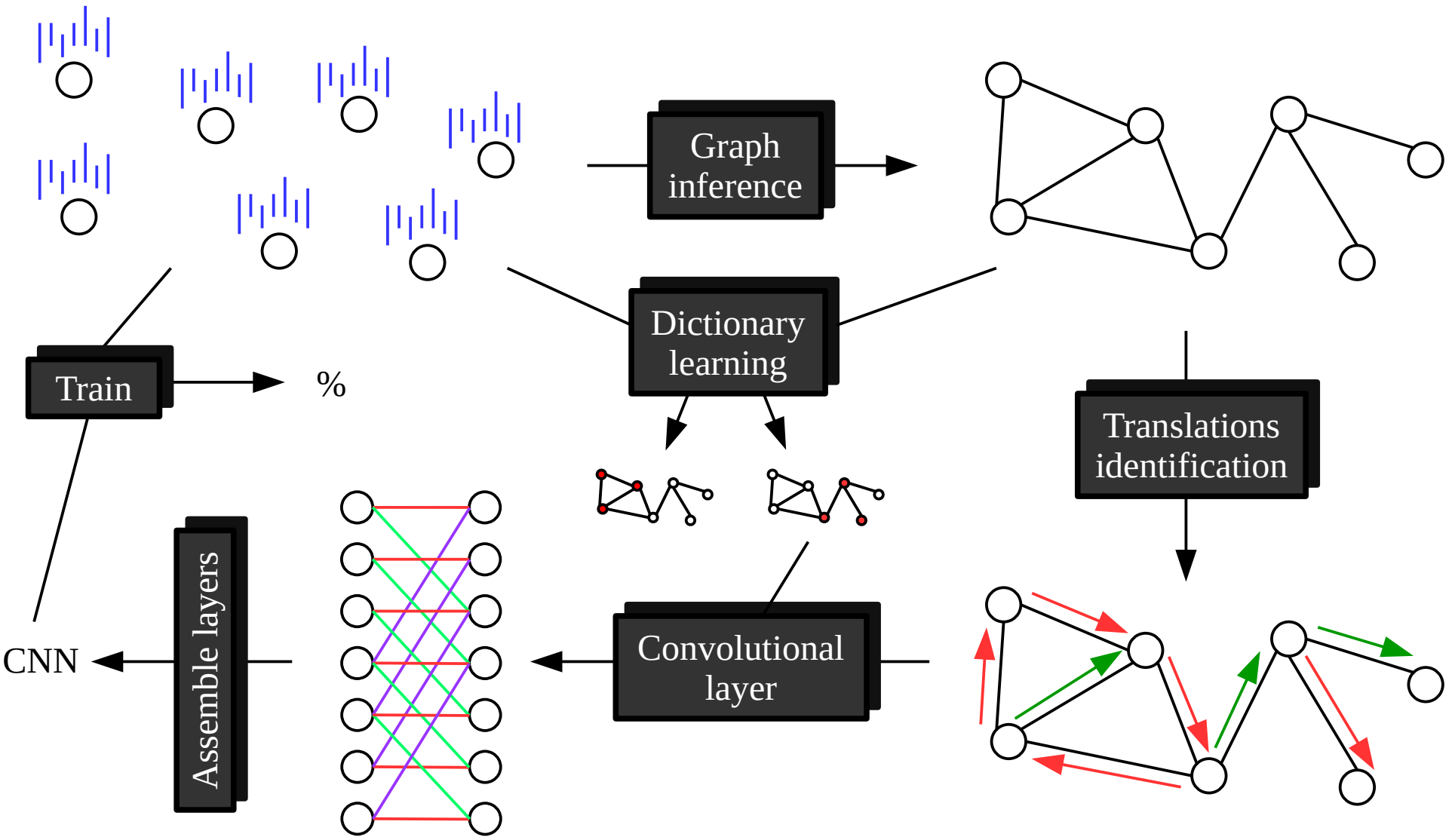
Supplementary — Dictionary learning — The idea



Supplementary — Dictionary learning — The idea



Supplementary — Dictionary learning — The idea

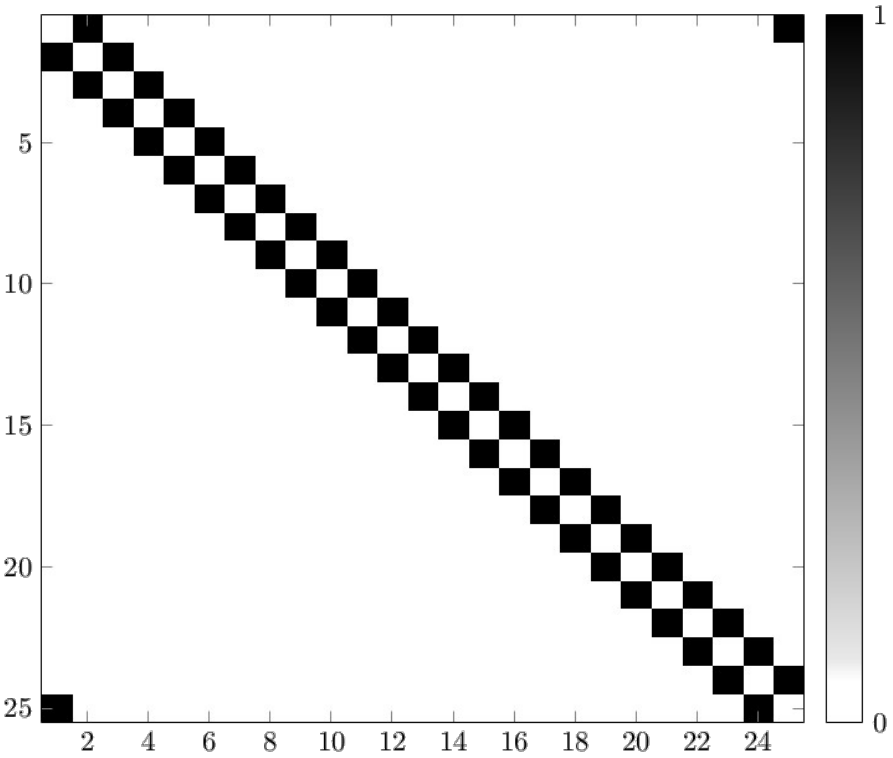


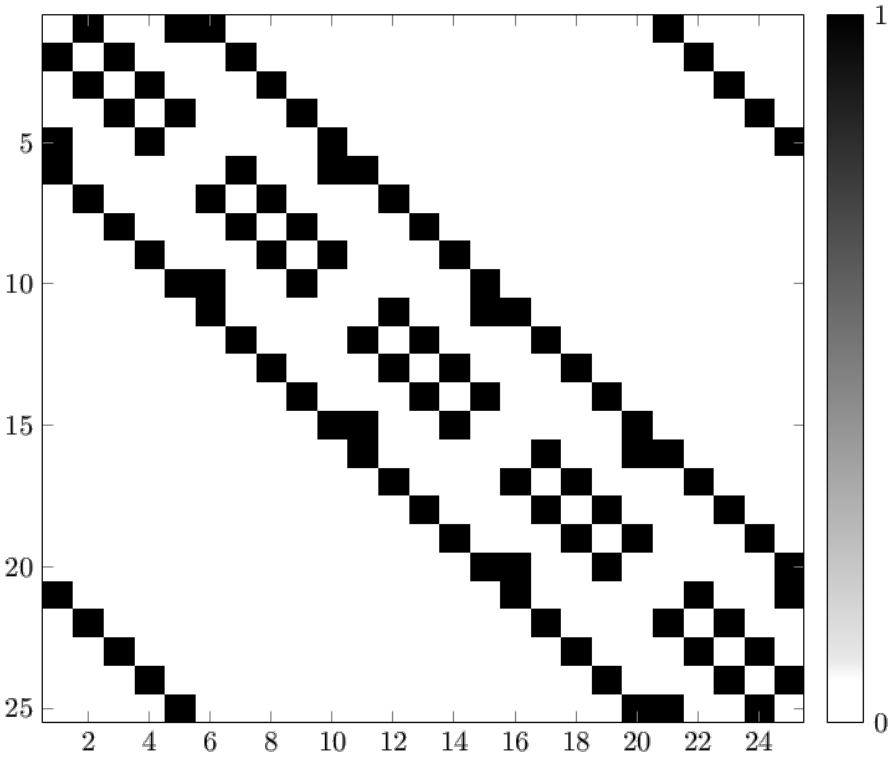
Outline

Supplementary slides

— Ideas on circulant matrices —

Supplementary — Circulant matrices — Particular graphs





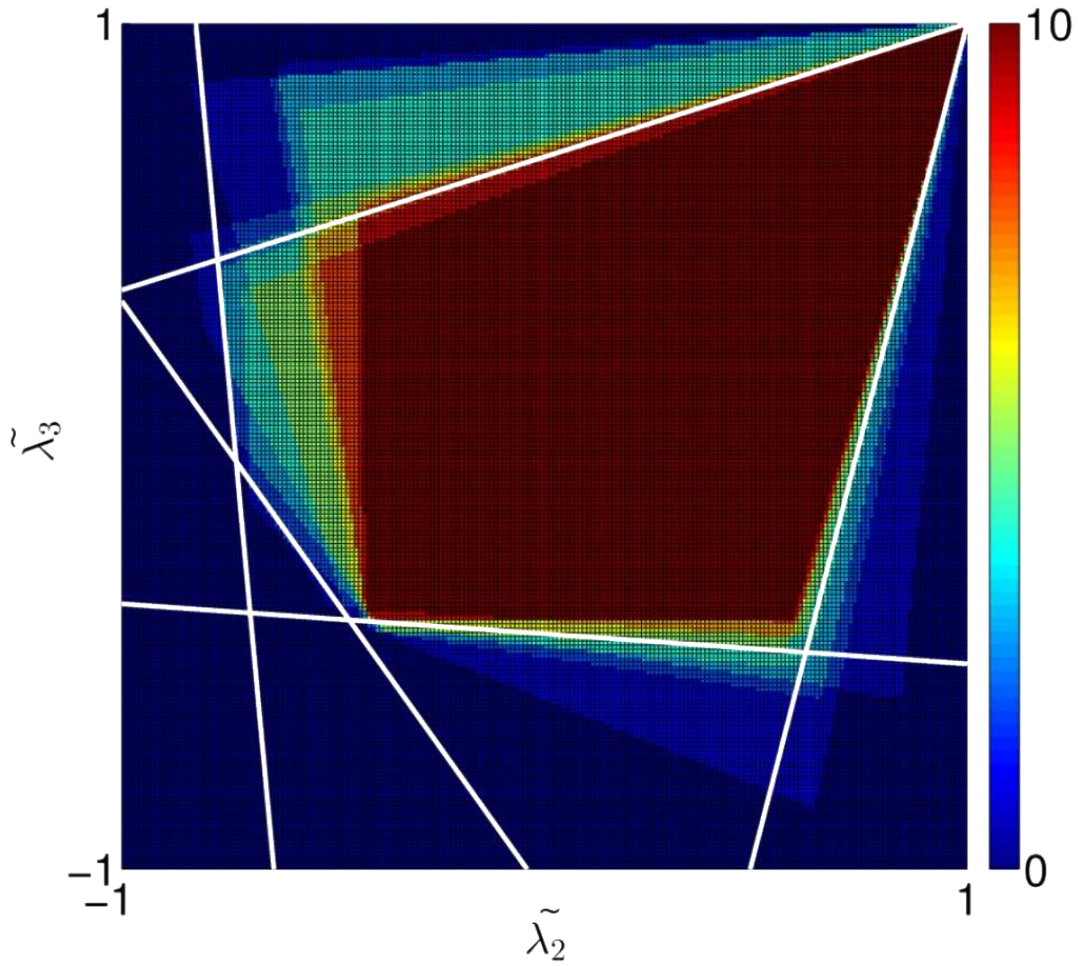
Outline

Supplementary slides

— Convergence of the eigenvectors of $\widetilde{\mathbf{\Sigma}}$ —

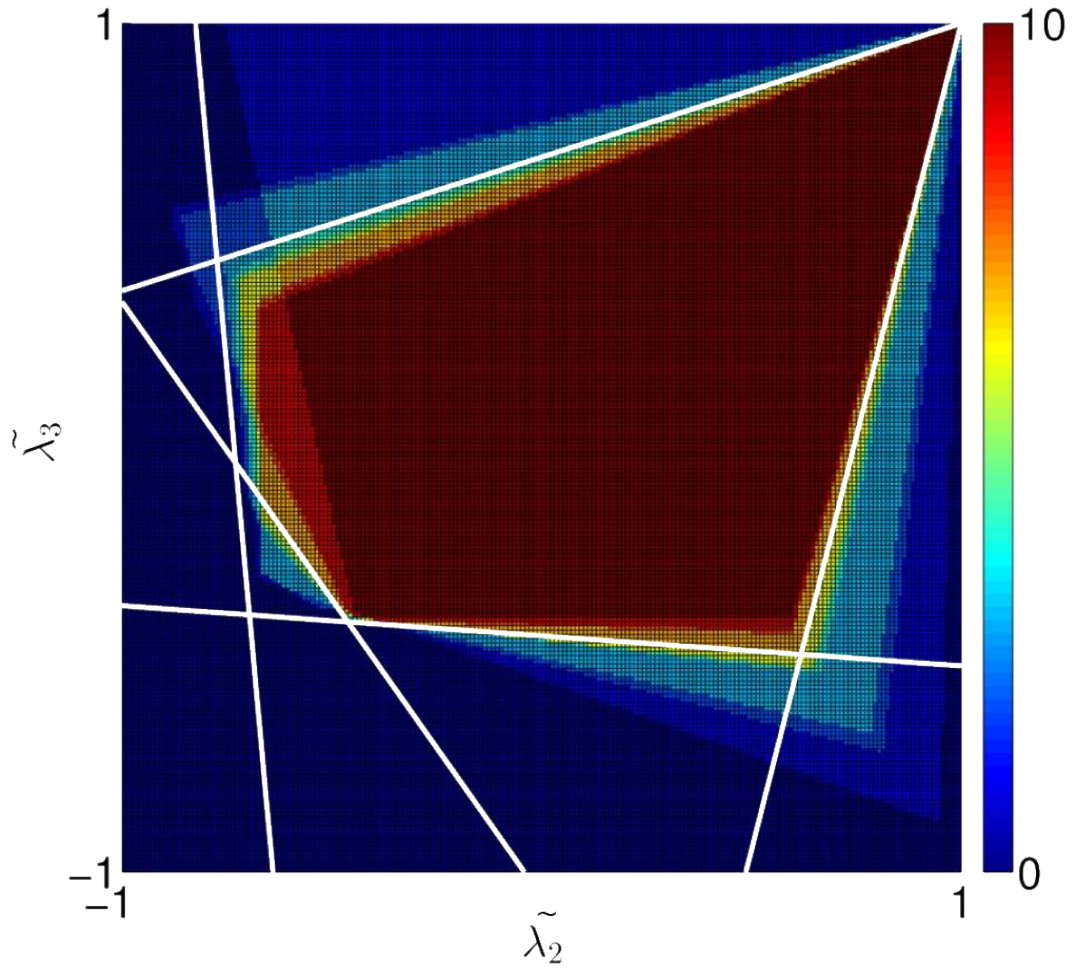
Supplementary — Eigenvectors of $\widetilde{\Sigma}$ — Experimentally

$$\mathbf{X} \xrightarrow[(10)]{} \widetilde{\Sigma} \longrightarrow \widetilde{\mathcal{X}} \longrightarrow$$



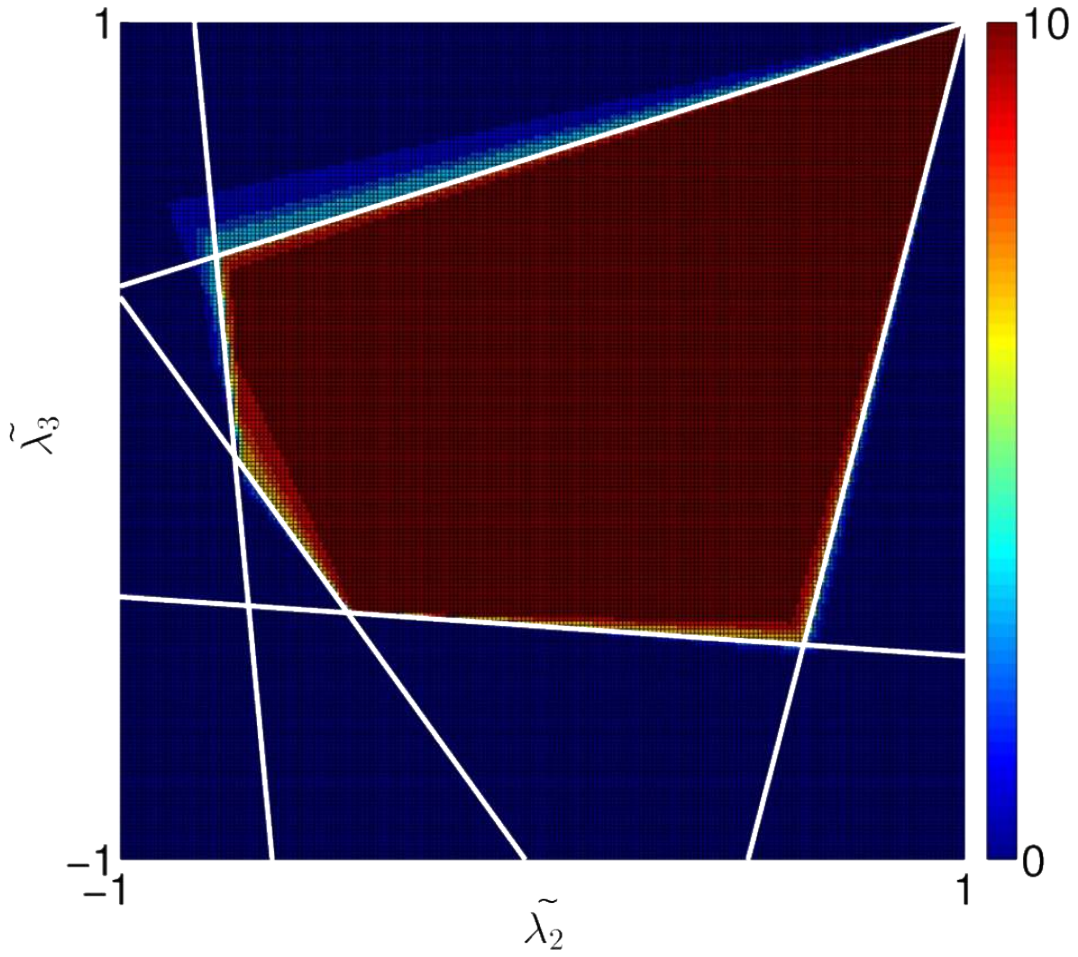
Supplementary — Eigenvectors of $\widetilde{\Sigma}$ — Experimentally

$$\mathbf{X} \xrightarrow[(100)]{} \widetilde{\Sigma} \longrightarrow \widetilde{\mathcal{X}} \longrightarrow$$



Supplementary — Eigenvectors of $\widetilde{\Sigma}$ — Experimentally

$$\mathbf{X} \xrightarrow[(1{,}000)]{} \widetilde{\Sigma} \longrightarrow \widetilde{\mathcal{X}} \longrightarrow$$



Supplementary — Eigenvectors of $\widetilde{\Sigma}$ — Formally

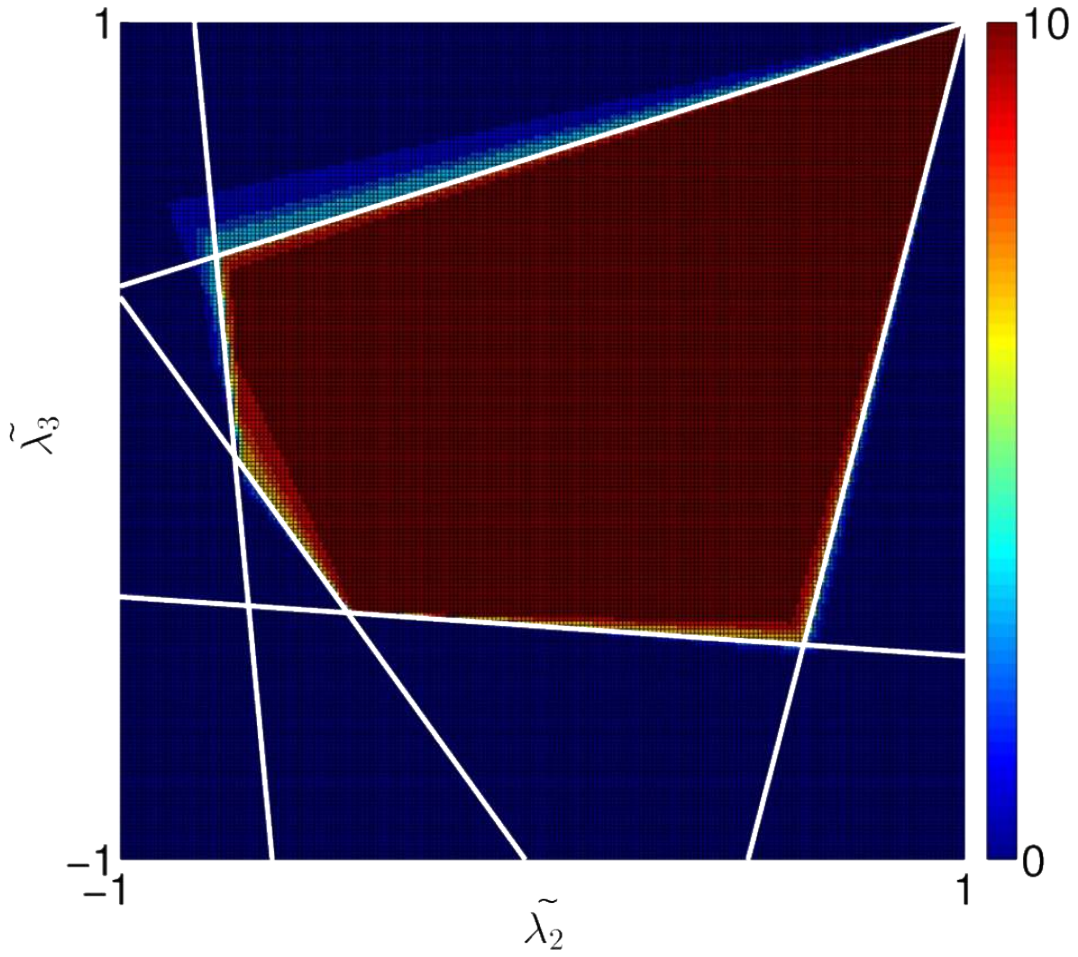
$$\mathbf{X} \xrightarrow[(1{,}000)]{} \widetilde{\Sigma} \longrightarrow \widetilde{\mathcal{X}} \longrightarrow$$

Variance of inner product between $\mathcal{X}_i$ and $\mathcal{X}_j$:

$$\frac{\lambda_i \widetilde{\lambda}_j}{(\lambda_i - \widetilde{\lambda}_j)^2}, \lambda_i \neq \lambda_j$$

Best estimator (for distinct λ):

$$\widetilde{\lambda_i^*} = \frac{M-1}{M}\widetilde{\lambda}_i$$

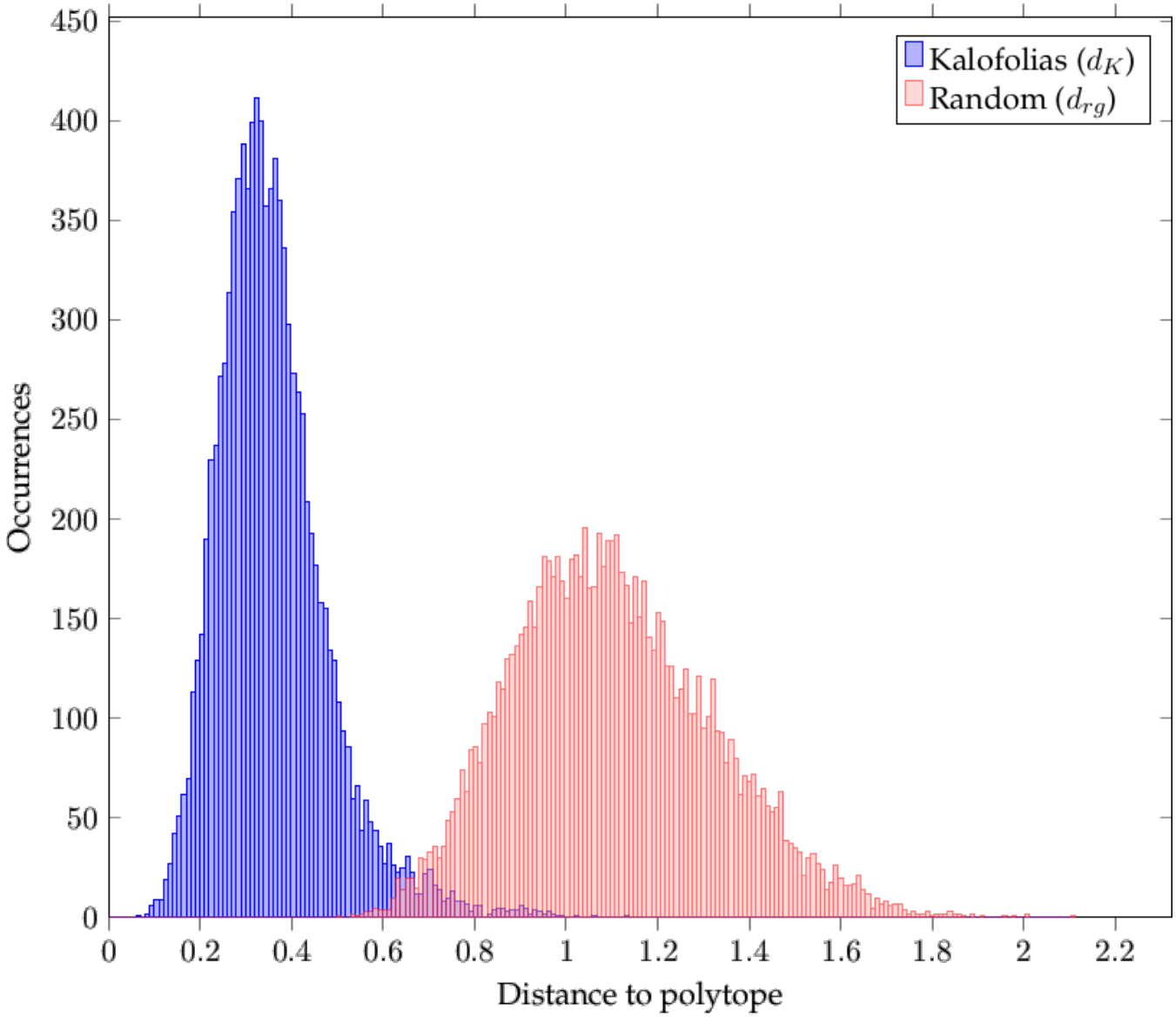


Outline

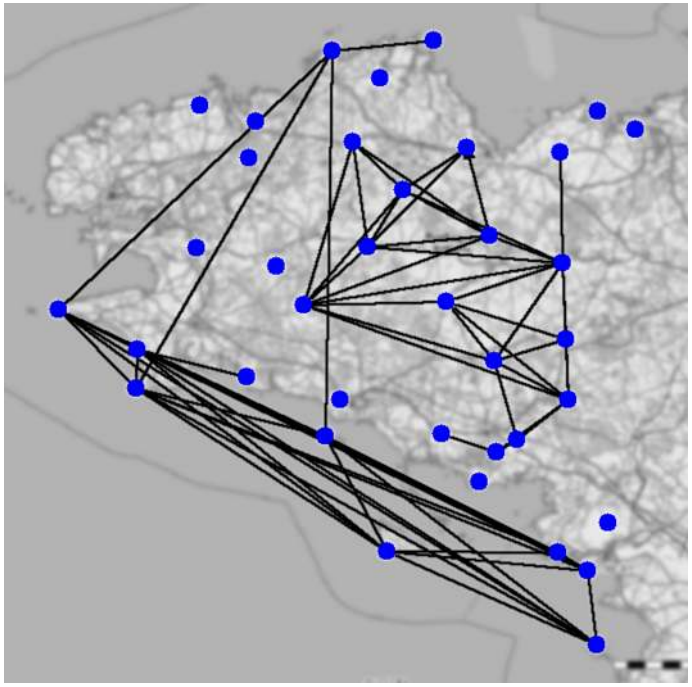
Supplementary slides

— Regularization of a graph inference method —

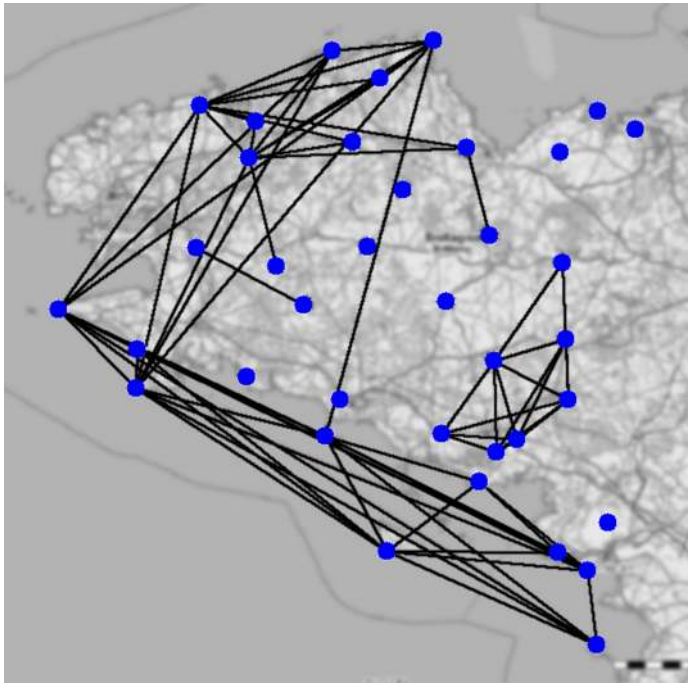
Supplementary — Regularization — Distance to the polytope



Supplementary — Regularization — Temperatures in Brittany



$\mathcal{G}_{\mathrm{K}}$



$\mathcal{G}_{\mathrm{K}}^{*}$