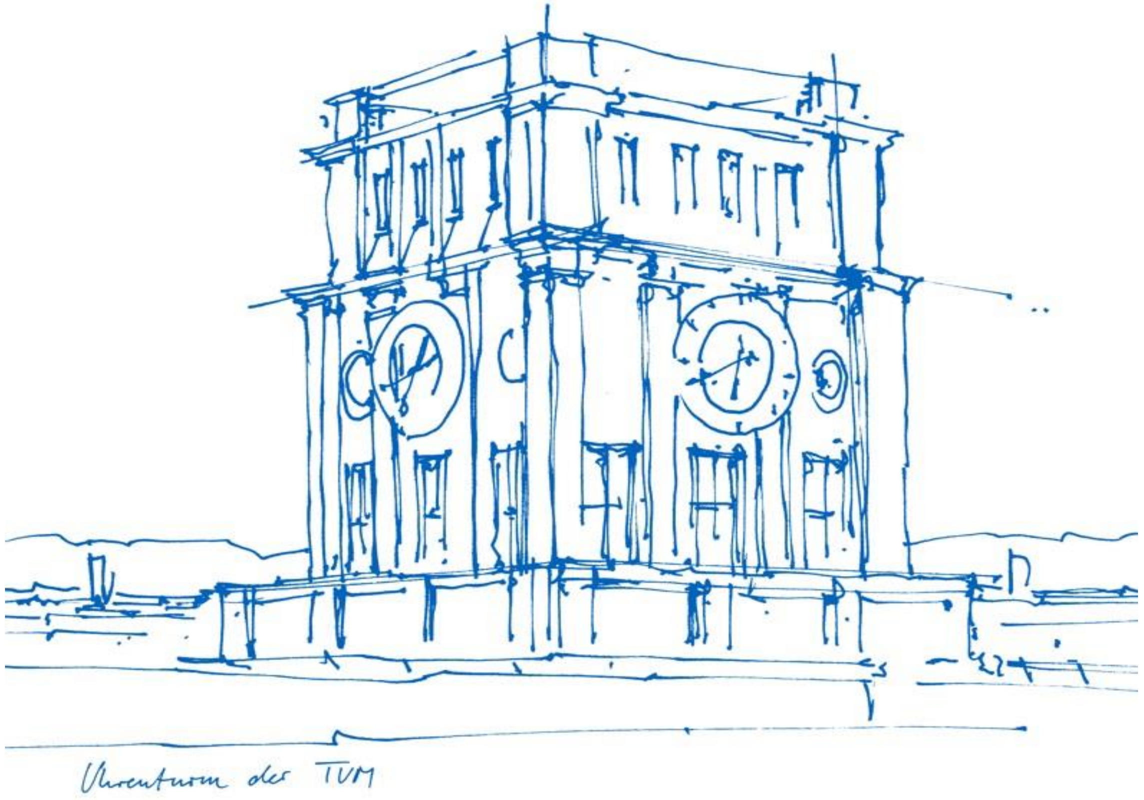
Chair of Communication Networks
Department of Electrical and Computer Engineering
Technical University of Munich

NeuroViNE: A Neural Preprocessor for Your Virtual Network Embedding Algorithm

Andreas Blenk* Patrick Kalmbach* Johannes Zerwas*
Michael Jarschel° Stefan Schmid^ Wolfgang Kellerer*

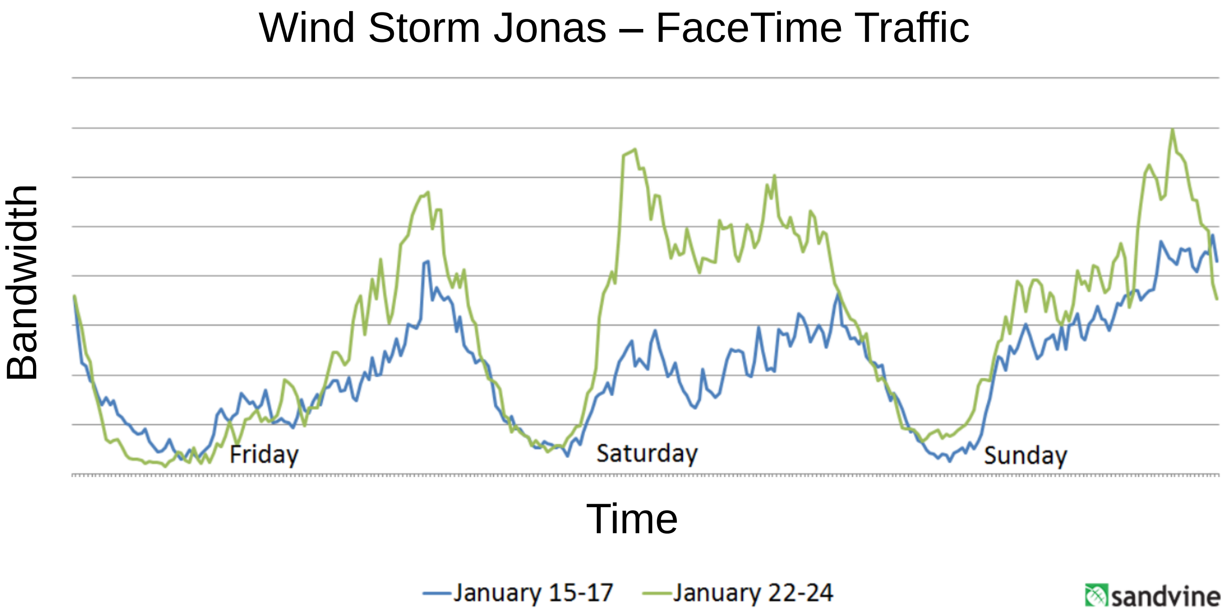
**Technical University of Munich, Germany*
°Nokia Bell Labs, Munich, Germany
^University of Vienna, Austria



Context

Virtual Networks Providing Predictable Performance

Increasing diversity of applications

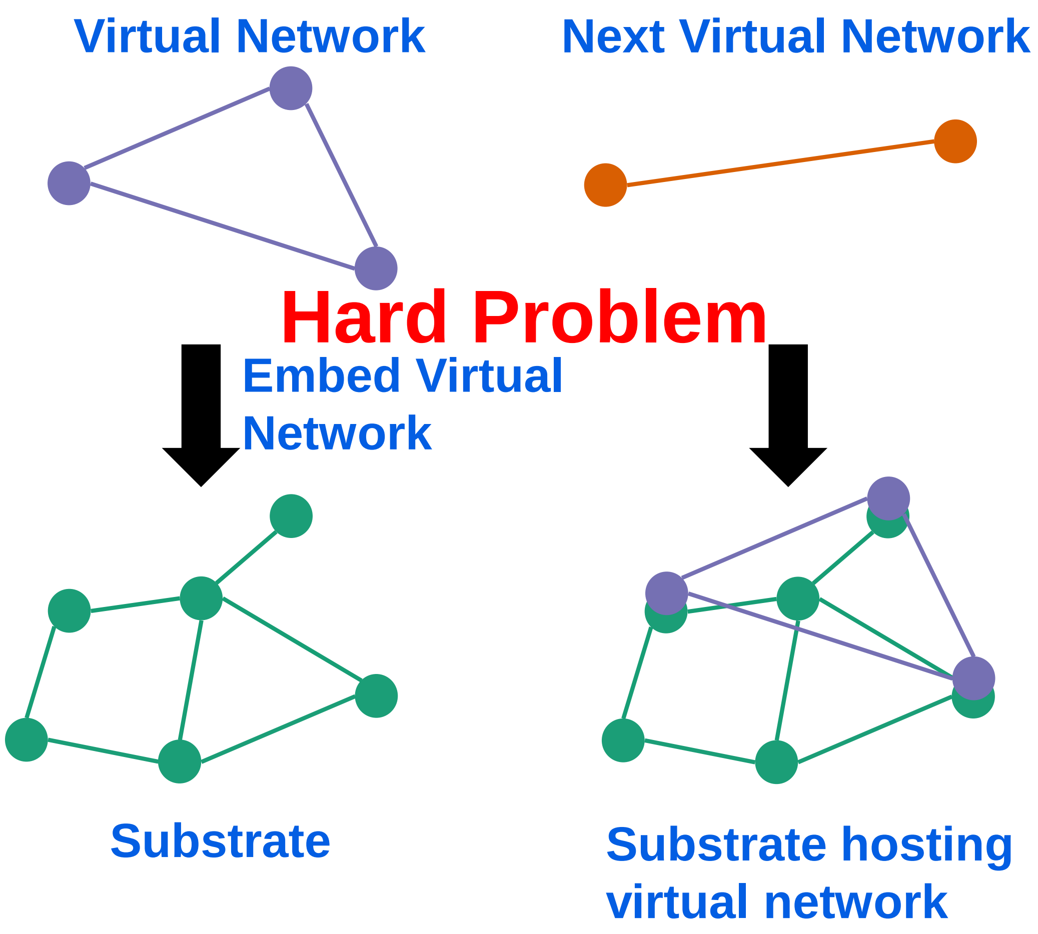


Frequently changing demands

- Network Virtualization: Resource sharing enables high and efficient network utilization
- Predictable application performance: Need for efficient performance isolation mechanisms

The Problem

Online Virtual Network Embedding (VNE) problem



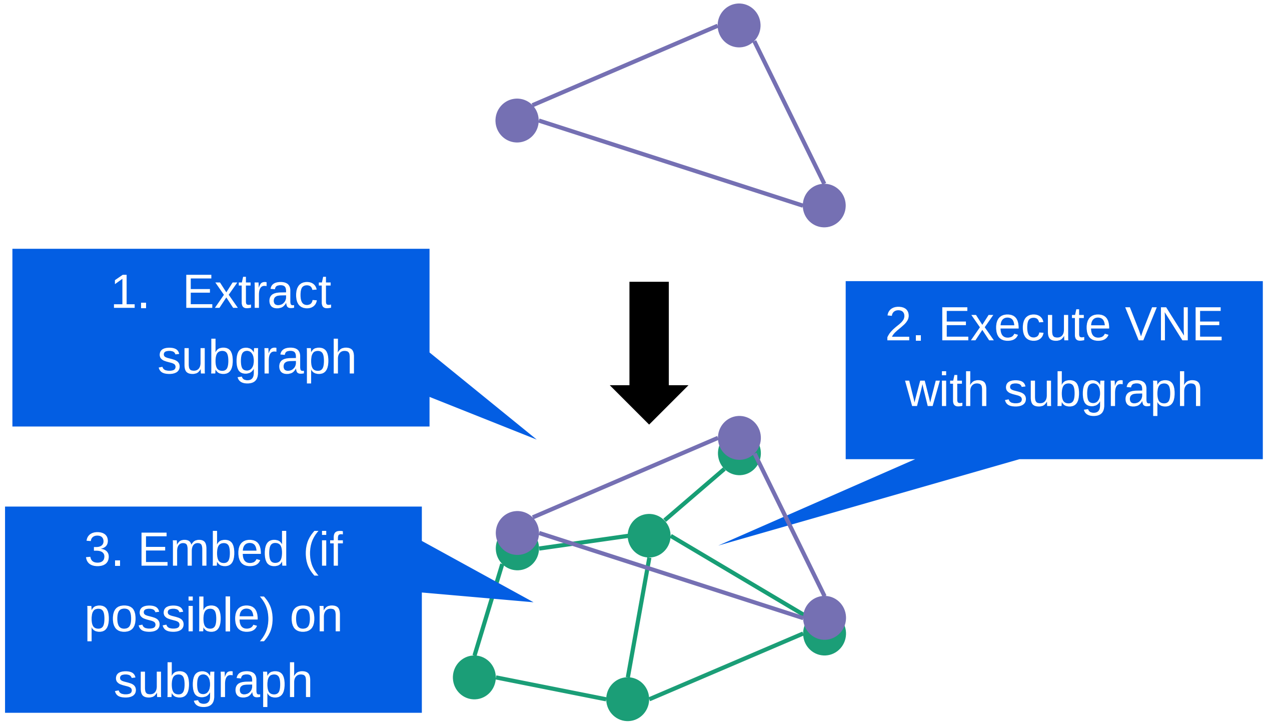
(1) Optimal solutions do not scale

Vs.

(2) Heuristics may result in large footprints

Neural Preprocessing to achieve (1) scalability and (2) quality

The Idea: Subgraph Extraction



→Reduce embedding cost of heuristics (search on close substrate nodes)

→Improve runtime of optimal algorithms (shrink search space)

But how do we find good subgraphs?!

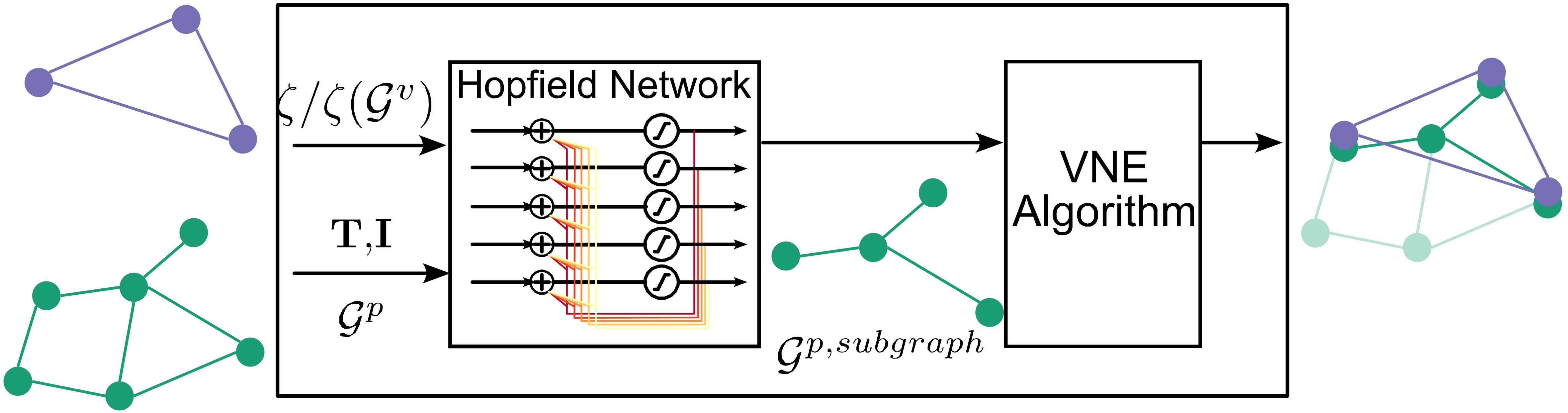
Contribution: NeuroViNE

Neural Computation

Parallel Computation

Implementable on hardware

Reuse existing VNE algorithms



Hopfield network solution provides nodes with high capacity close to each other

Neural Computation of Decisions in Optimization Problems

“**Neural**” computation of decisions in **optimization** problems

JJ **Hopfield**, DW Tank - Biological cybernetics, 1985 - Springer

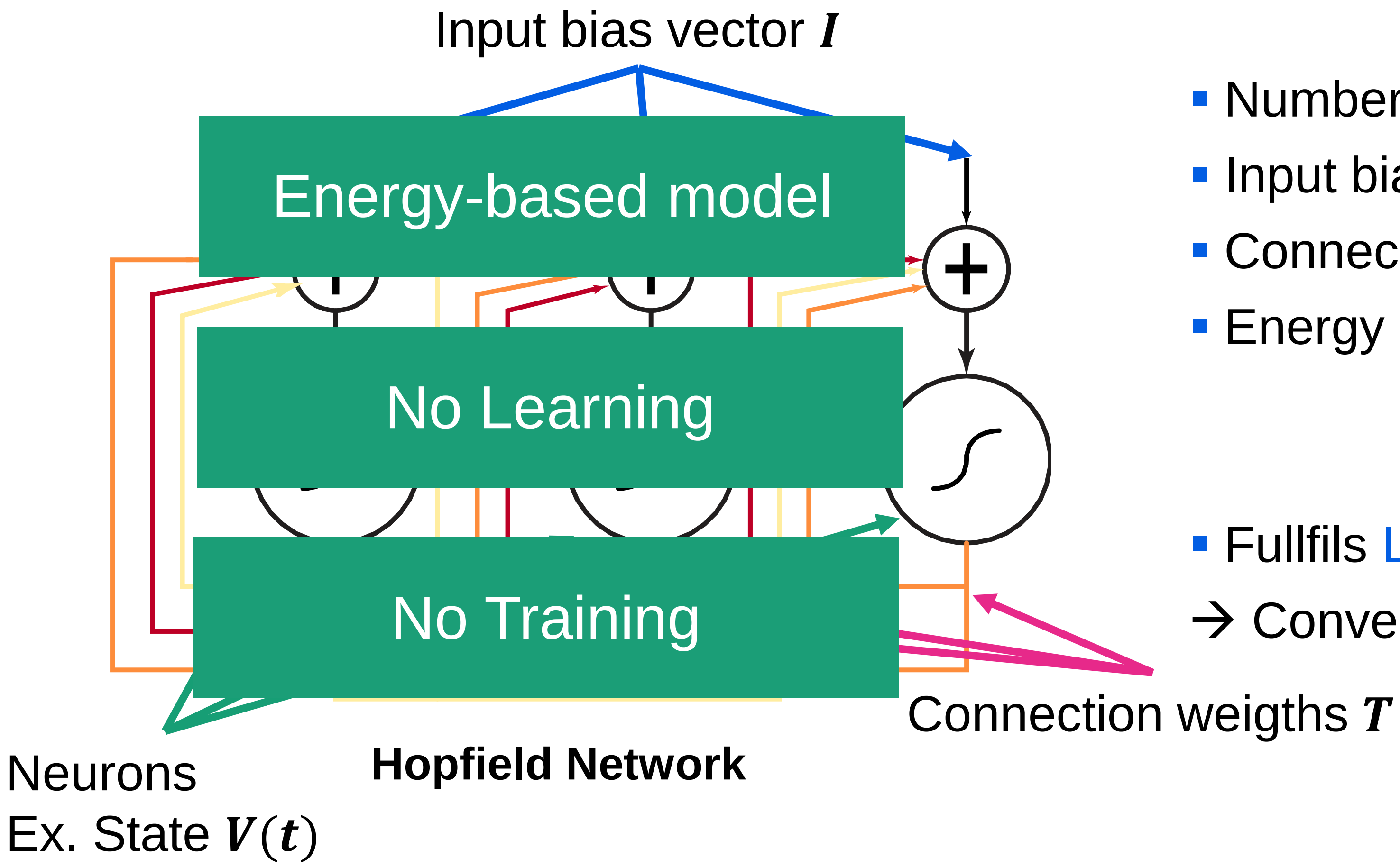
Abstract Highly-interconnected networks of nonlinear analog neurons are shown to be extremely effective in computing. The networks can rapidly provide a collectively-computed solution (a digital output) to a problem on the basis of analog input information. The …

Zitiert von: 7329 Ähnliche Artikel Alle 33 Versionen

John Hopfield

Hopfield Network

An Artificial Recurrent Neural Network (which can be used for optimization)



Hopfield Network

- Number of neurons
- Input bias vector $\boldsymbol{I}$
- Connection weigths $\boldsymbol{T}$
- Energy of network

$$E = -\frac{1}{2}\boldsymbol{V}^T\boldsymbol{T}\boldsymbol{V} - \boldsymbol{V}^T\boldsymbol{I}$$

- Fullfils Lyapunov function property

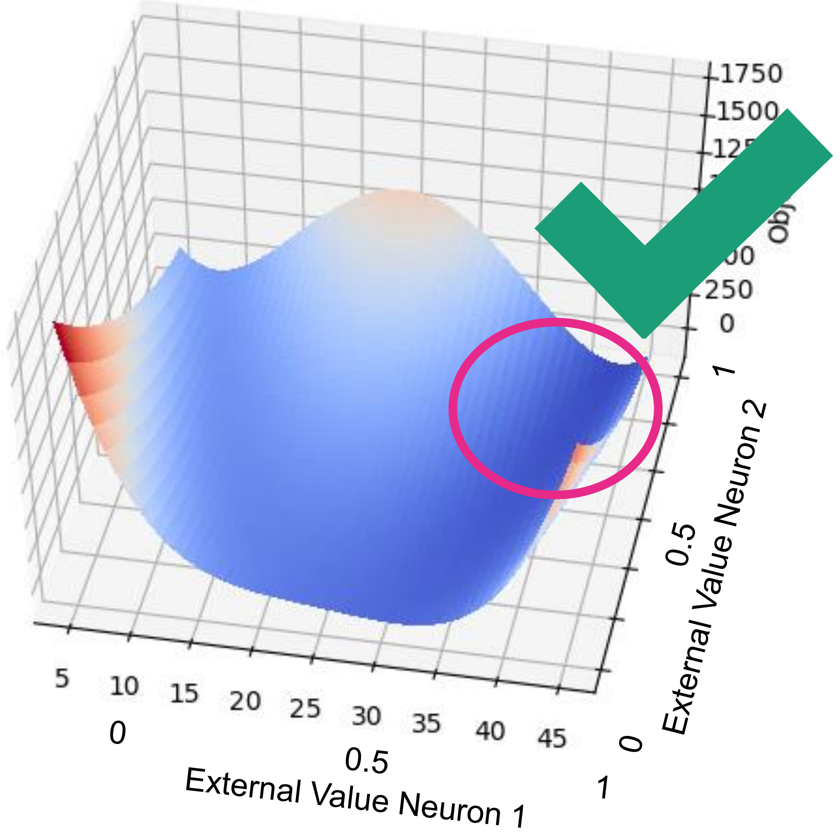
→ Convergence to local (global) optima guaranteed

Hopfield Network Properties

How to map Virtual Network Embedding problem?

Hopfield Network

How to use for optimization ...



1. Optimization problem: find subgraph with low resource footprint and high probability for accepting virtual network
2. VNE problem energy function

   $E = \boldsymbol{V}^T\left(\boldsymbol{\Psi}(\mathrm{t}) + \alpha \cdot \mathbf{T}^{\mathrm{constraint}}\right)\boldsymbol{V} + \boldsymbol{V}^T\left(\boldsymbol{\Xi}(\mathrm{t}) + \alpha \cdot \mathbf{I}^{\mathrm{constraint}}\right)$

3. Derive: $\boldsymbol{\Psi}(\mathrm{t})$, $\mathbf{T}^{\mathrm{constraint}}$, $\boldsymbol{\Xi}(\mathrm{t})$, $\mathbf{I}^{\mathrm{constraint}}$
4. Execute network: solve
5. After exectution → Neuron states (values) indicate subgraph nodes

Hopfield Optimization Procedure

We do not solve VNE directly ...
But show Hopfield‘s preprocessing capabilities

NeuroViNE‘s Hopfield Network Energy Function

Select paths
with low costs =
low energy

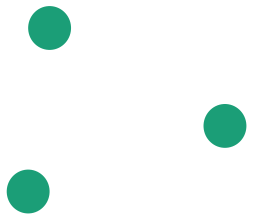
Satisfying
constraint =
low energy

Select virtual nodes
with high CPU ratio =
low energy

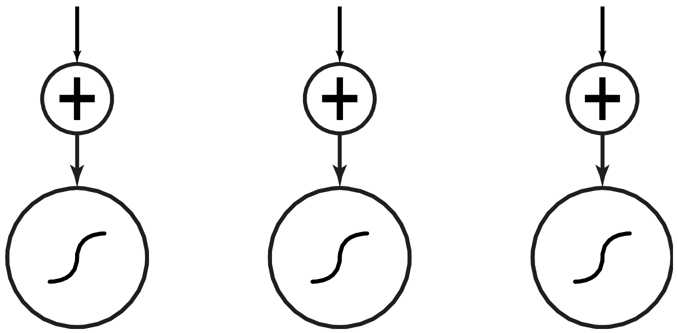
$$E = \boldsymbol{V}^T\left(\boldsymbol{\Psi}(\mathrm{t}) + \alpha \cdot \mathbf{T}^{\mathrm{constraint}}\right)\boldsymbol{V} + \boldsymbol{V}^T\left(\boldsymbol{\Xi}(\mathrm{t}) + \alpha \cdot \mathbf{I}^{\mathrm{constraint}}\right)$$

NeuroViNE‘s Hopfield Network Construction

Example for 3-Node Substrate and 2-Node Virtual Network



$\boldsymbol{\Xi_{\mathrm{i}}(t)}$



3 substrate nodes with CPU resource

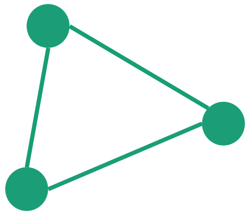
3 neurons - Input bias vector considers CPU

Node ranking

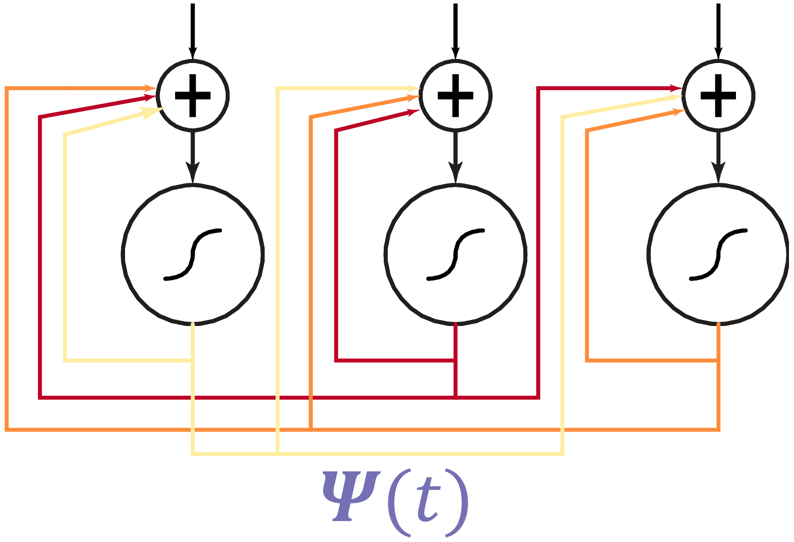
$$\Xi_{\mathrm{i}}(t) = \frac{\max_{N_j \in \mathcal{N}} C_j(t) - C_i(t)}{\max_{N_j \in \mathcal{N}} C_j(t)}$$

Path Ranking

NeuroViNE's Hopfield Network Construction



3 links with datarate attributes



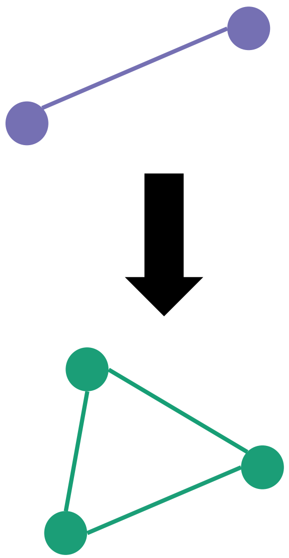
3 times 3 entries of weight matrix

Path ranking

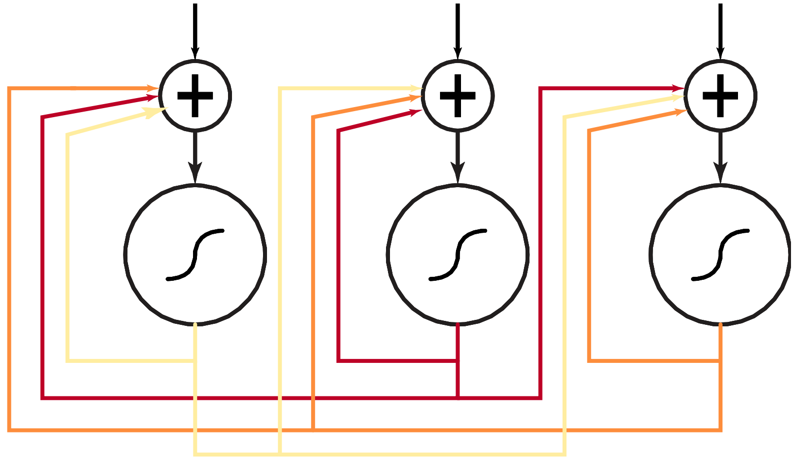
$$\Psi_{ij}(t) = \gamma \frac{D_{ij}(t)}{\max_{ij}(D(t))}$$

Keeping Constraints

NeuroViNE‘s Hopfield Network Construction



2 Virtual nodes



2 out of 3 neurons should be chosen

Node number selection constraints

$$T_{ij}^{constraint} = \begin{cases} 1 \; if \; i \neq j \\ 0 \; otherwise \end{cases}$$

$$I_k^{constraint} = -(2 \cdot \zeta - 1)$$

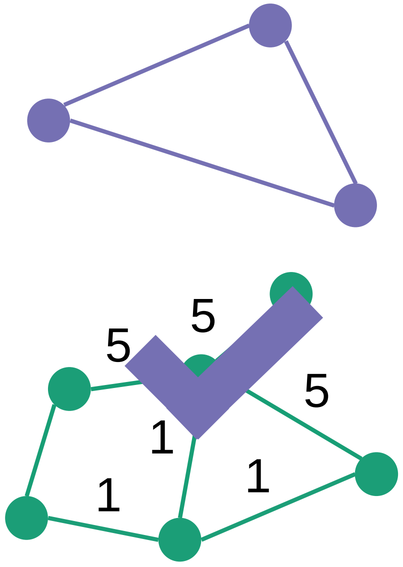
NeuroViNE‘s Hopfield Network Energy Function

$$E = \boldsymbol{V}^T\left(\boldsymbol{\Psi}(\mathrm{t}) + \alpha \cdot \mathbf{T}^{\mathrm{constraint}}\right)\boldsymbol{V} + \boldsymbol{V}^T\left(\boldsymbol{\Xi}(\mathrm{t}) + \alpha \cdot \mathbf{I}^{\mathrm{constraint}}\right)$$

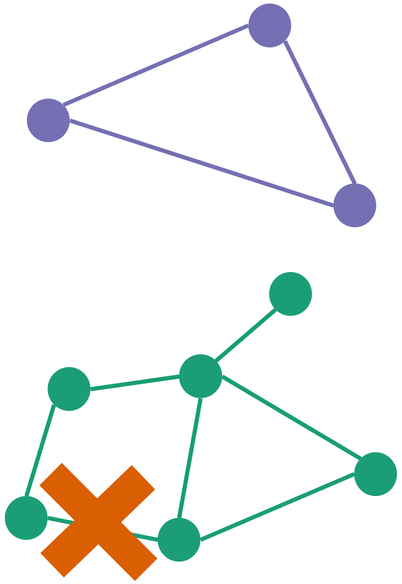
Select paths with low costs = low energy

Satisfying constraint = low energy

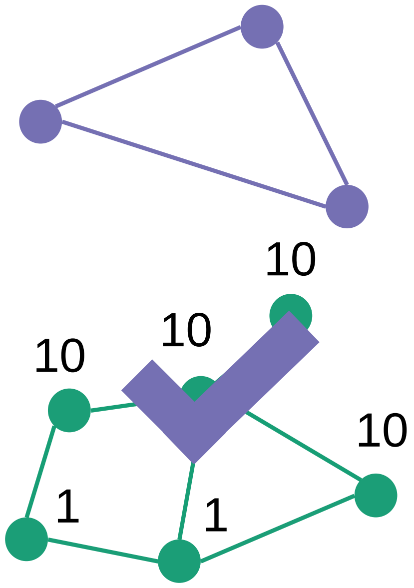
Select virtual nodes with high CPU ratio = low energy



Low Energy



High Energy



Low Energy

NeuroViNE Evaluation

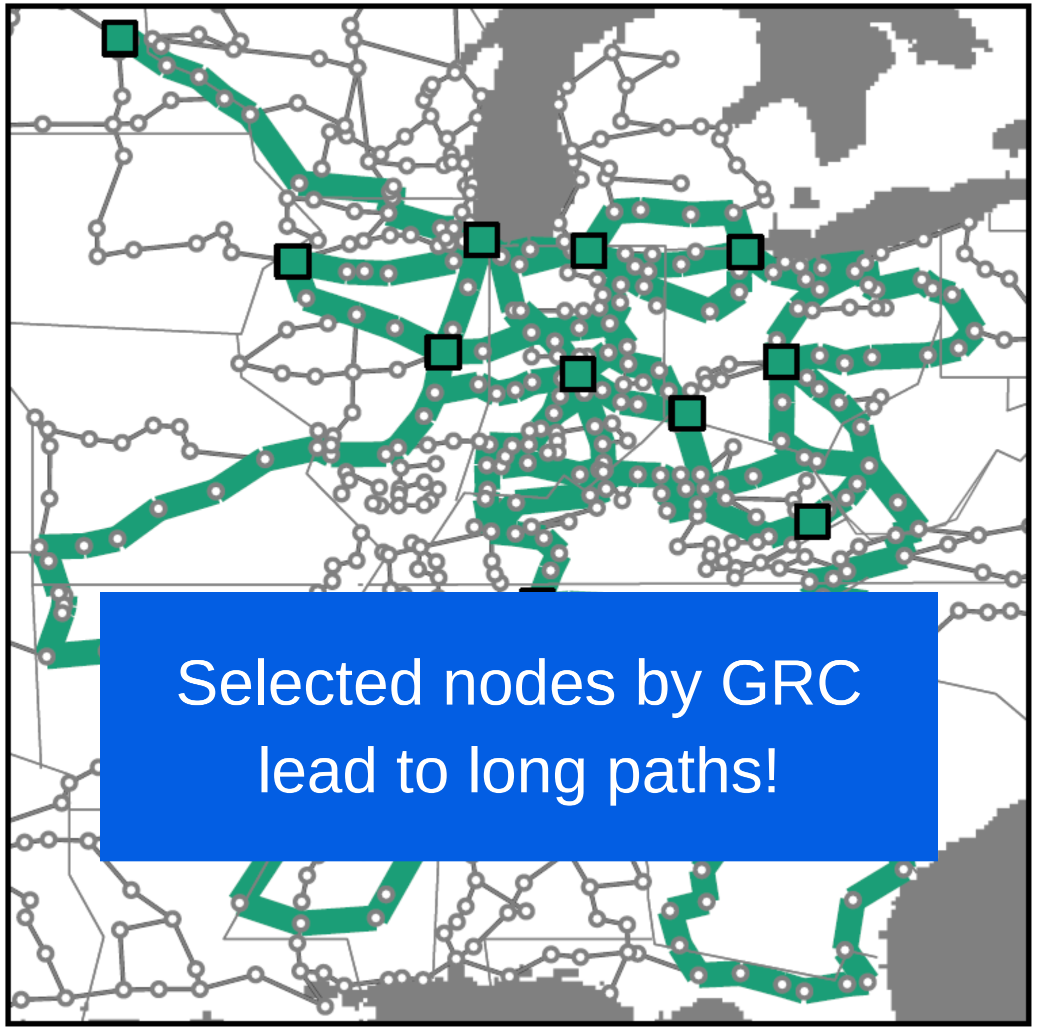
- NeuroViNE in combination with state-of-the-art VNE algorithms: GRC, ViNEYard, Shortest Distance Path (SDP: optimal algorithm with limited solving time)

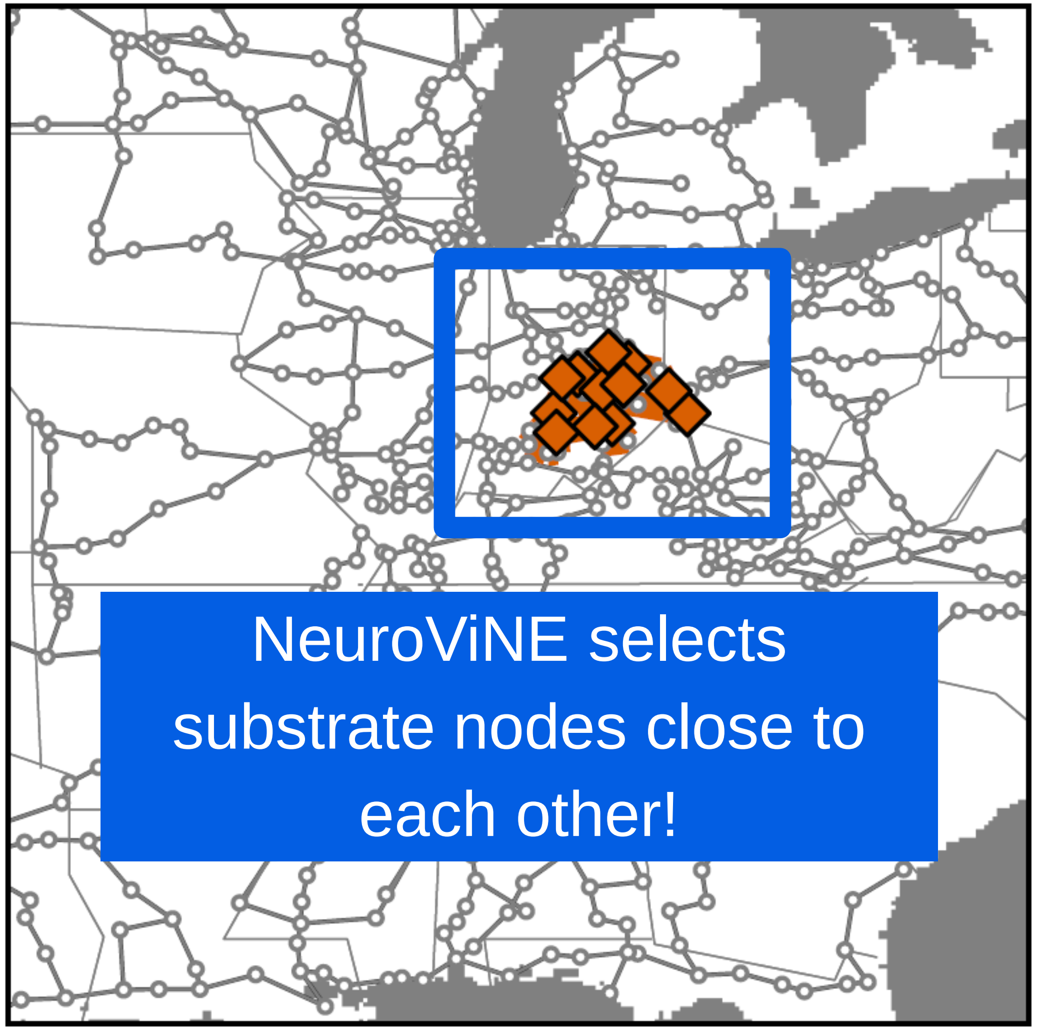
- Substrate topologies: random network topologies, wide area networks and datacenter topologies (FatTree and Bcube)

- Performance Measures: Acceptance ratio, revenue-cost-ratio, total revenue, algorithm modeling time (for optimal algorithms), algorithm solving time (for optimal alorithms)

- Simulation settings: at least 2500 VNs per topology with different arrival rates

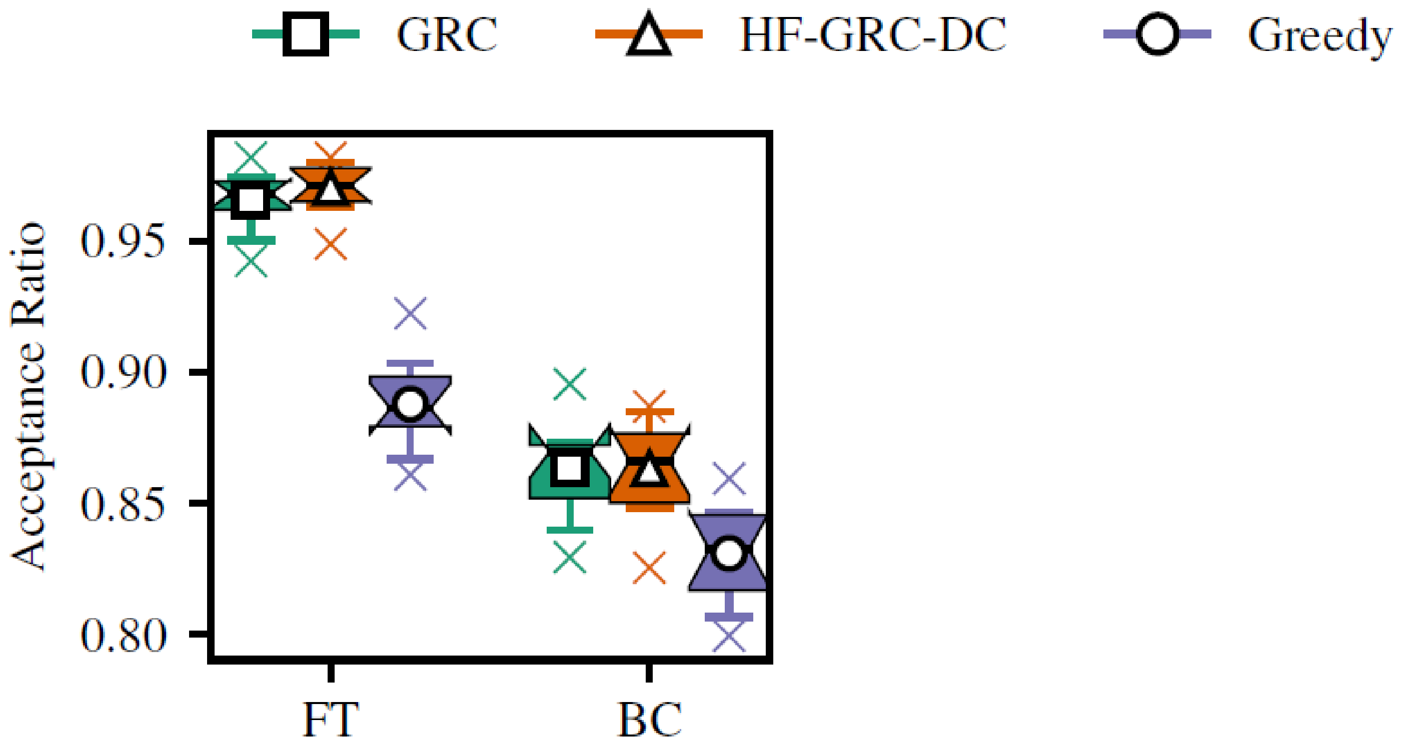
NeuroViNE: An Illustrative Example for GRC on 750 nodes ISP network





NeuroViNE: Efficient also in Datacenters

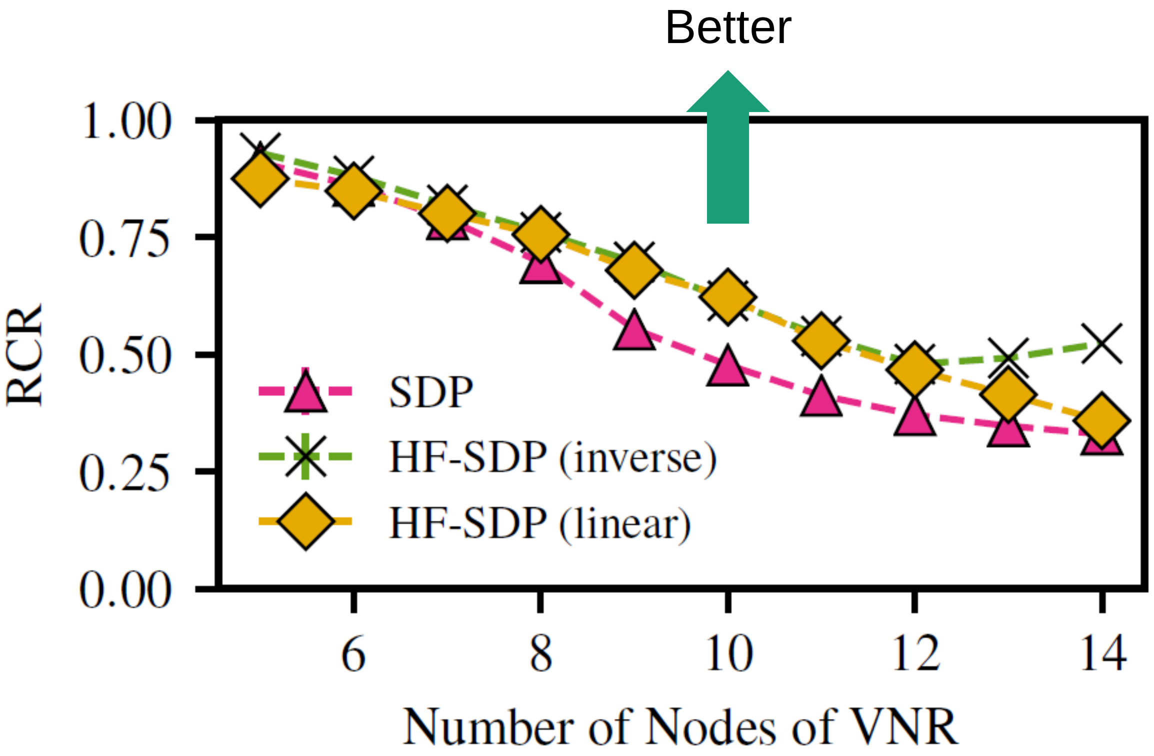
Uses a datacenter modifcation (see paper)

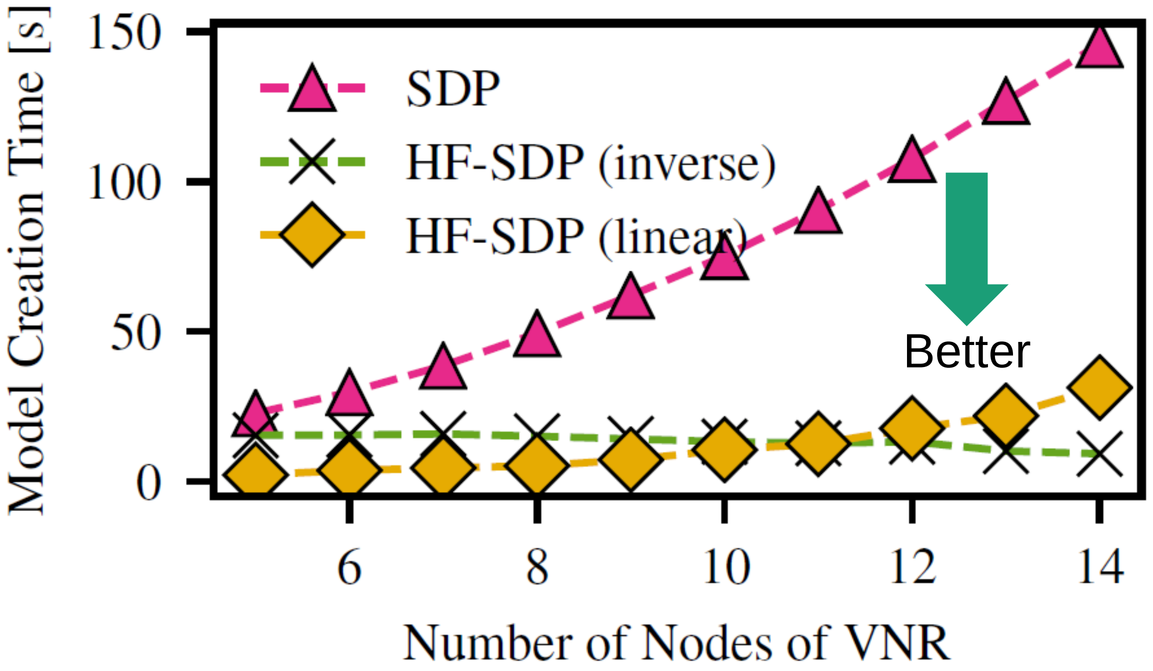


NeuroViNE shows similar acceptance ratios

... but saves cost

NeuroViNE Helps Optimal Algorithms to Become Credible Alternatives





Preselected nodes improve revenue-cost-ratio (RCR)

Subgraph reduces variables when modeling/solving

Other Ways to Improve Networking Algorithms (Our related work)

Blenk, Andreas; Kalmbach, Patrick; Schmid, Stefan; Kellerer, Wolfgang: o'zapft is: Tap Your Network Algorithm's Big Data! ACM SIGCOMM 2017 Big-DAMA Workshop, 2017

This talk: NeuroViNE

Neural Computation

Machine Learning

He, Mu; Kalmbach, Patrick; Blenk, Andreas; Schmid, Stefan; Kellerer, Wolfgang: Algorithm-Data Driven Optimization of Adaptive Communication Networks. IEEE ICNP Workshop on Machine Learning and Artificial Intelligence in Computer Networks, 2017

Speed up networking algorithms
&
Improve algorithm soluation quality

Blenk, Andreas; Kalmbach, Patrick; van der Smagt, Patrick; Kellerer, Wolfgang: Boost Online Virtual Network Embedding: Using Neural Networks for Admission Control. CNSM, 2016

Alpha GO inspired (Monte Carlo Tree Search)

Probabilistic Modeling

Johannes Zerwas, Patrick Kalmbach, Carlo Fuerst, Arne Ludwig, Andreas Blenk, Wolfgang Kellerer, and Stefan Schmid. “Ahab: Data-Driven Virtual Cluster Hunting.” In: Proc. IFIP Networking. accepted for publication. Zurich, Switzerland, May 2018, 1–9.

Patrick Kalmbach, Andreas Blenk, Wolfgang Kellerer, and Stefan Schmid. “Themis: A Data-Driven Approach to Bot Detection (Short Abstract).” In: Proc. IEEE INFOCOM. accepted for publication. Honolulu, HI, USA, 2018, 1–2.

Conclusion

- Subgraph extraction targeting Online Virtual Network Embedding problem
- Designed Hopfield network for subgraph extraction (neural computation, fast!)
- Improved efficiency of existing VNE algorithms: reduced cost and speed up
- Opening interesting future work: energy-based models, automated configuration parameter tuning, restricted boltzmann machines, ...

Thank you!

&

Questions?