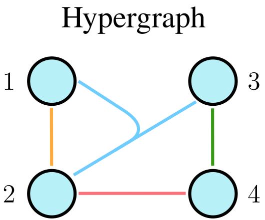
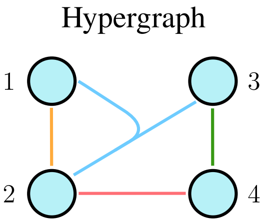
HYGENE: A Diffusion-based Hypergraph Generation Method

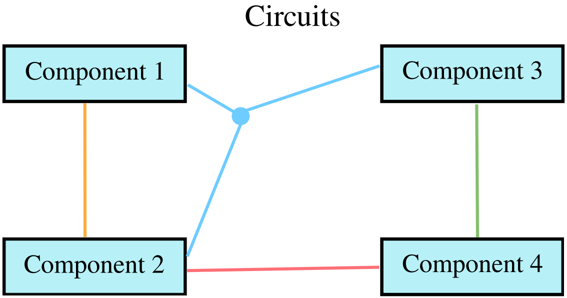
Dorian Gailhard (PhD student).
dorian.gailhard@telecom-paris.fr

What's a hypergraph and why it's useful?



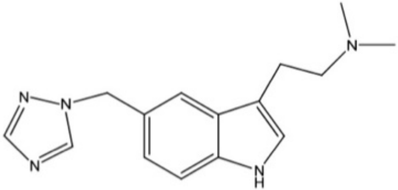
What's a hypergraph and why it's useful?





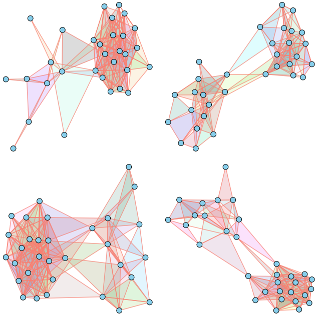
- Hyperedges group components connected to the same router.

What's a hypergraph and why it's useful?

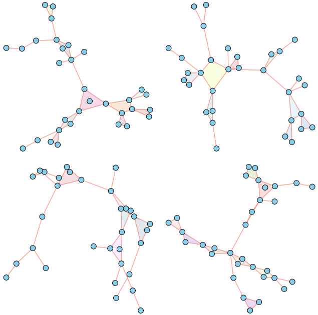


Hyperedges could group atoms belonging to the same compound (for example the three nitrogen atoms on the left side) in order to highlight that their relationship is not on a two-by-two basis but on a higher-order scale?

Objective



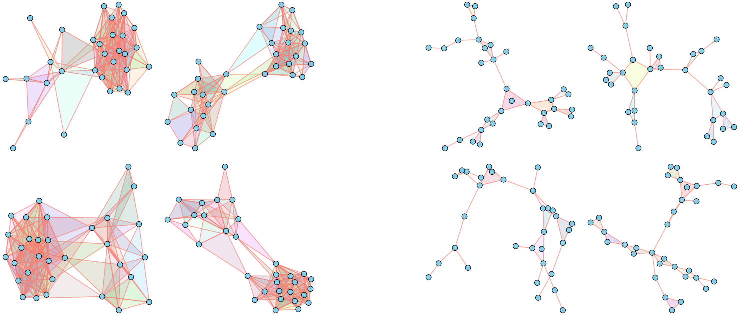
(a) Stochastic blocks model



(b) Tree hypergraphs

- Given a dataset of hypergraphs sharing the same properties, train a model able to generate additional samples.

Objective

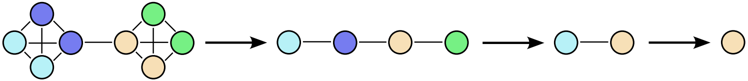


(a) Stochastic blocks model (b) Tree hypergraphs

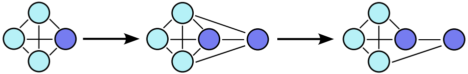
- Given a dataset of hypergraphs sharing the same properties, train a model able to generate additional samples.
- In this paper we only consider featureless hypergraphs.

Approach: iterative local expansion

- Based on *Efficient and Scalable Graph Generation through Iterative Local Expansion* by Bergmeister, A., Martinkus, K., Perraudin, N. and Wattenhofer, R. (ICLR 2024).



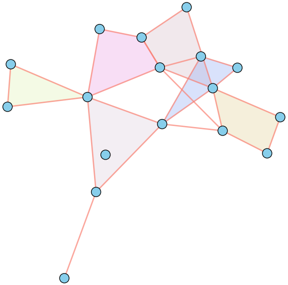
Reduce the graph by merging pairs of adjacent nodes



Learn to reconstruct the graph

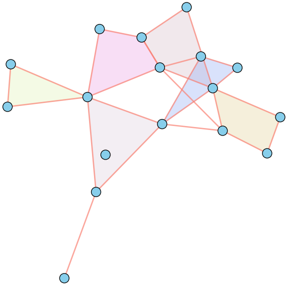
- The generalization of this approach to hypergraphs is not straightforward

Hypergraph projections we used

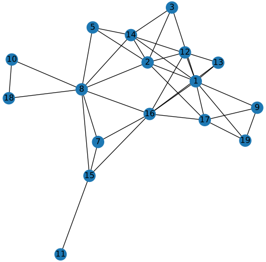


(a) Original hypergraph

Hypergraph projections we used



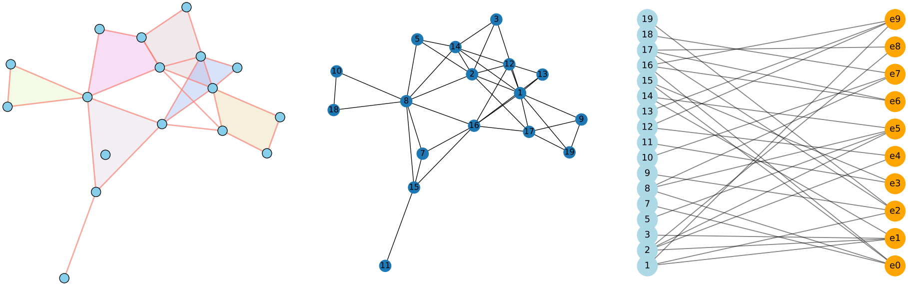
(a) Original hypergraph



(b) Clique expansion

- **Clique expansion:** Hyperedges are collapsed into cliques (a regular edge connects all pairs of nodes). It can be weighted.

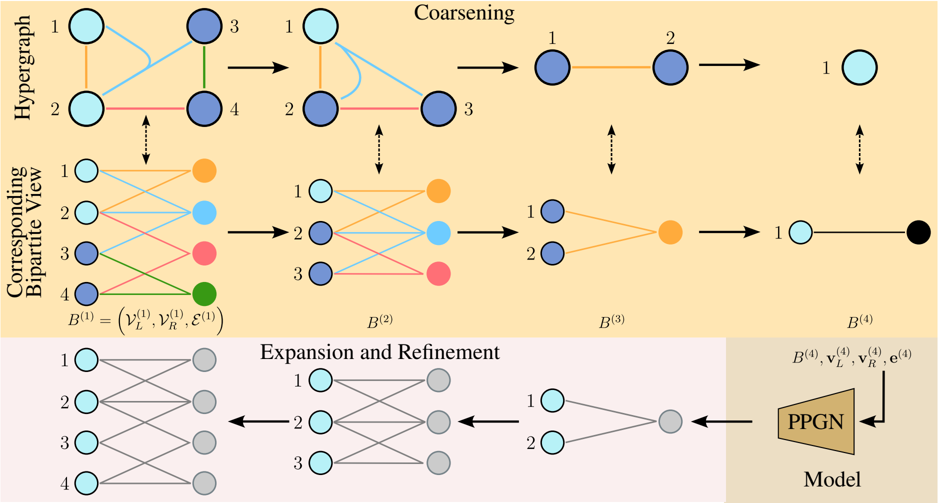
Hypergraph projections we used



(a) Original hypergraph (b) Clique expansion (c) Star expansion

- **Clique expansion:** Hyperedges are collapsed into cliques (a regular edge connects all pairs of nodes). It can be weighted.
- **Star expansion:** One side corresponds to nodes, the other to hyperedges, and each hyperedge is connected to all its nodes.

What we did?

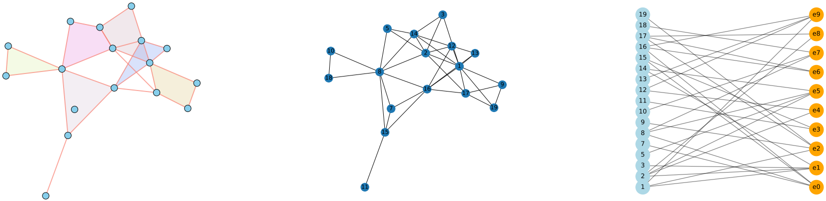


Coarsening
Hypergraph
Corresponding Bipartite View
$B^{(1)} = \left(\mathcal{V}_L^{(1)}, \mathcal{V}_R^{(1)}, \mathcal{E}^{(1)}\right)$
$B^{(2)}$
$B^{(3)}$
$B^{(4)}$
Expansion and Refinement
$B^{(4)}, \mathbf{v}_L^{(4)}, \mathbf{v}_R^{(4)}, \mathbf{e}^{(4)}$
PPGN
Model

Why not use conventional graph generation models?

- Using graph generators to produce hypergraph representations is infeasible because:
  1. Generating the clique expansion and recovering the associated hypergraph is an NP-hard problem as it requires the enumeration of all cliques
  2. For the bipartite representation, determining which side corresponds to nodes and which to hyperedges is non-trivial
- Besides, we empirically observe that graph-based models often struggle to generate valid bipartite graphs. For example, we only obtained 30% of correct bipartite graphs for both models by Bergmeister *et al.* (2024) and Vignac *et al.* (2023).

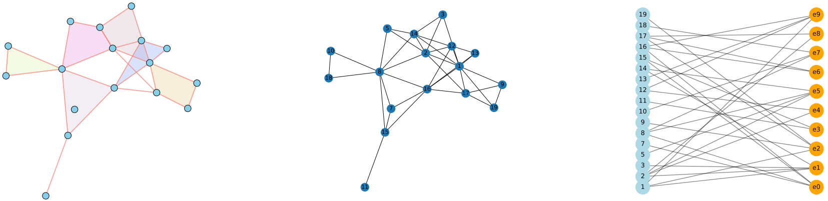
Why different projections?



(a) Original hypergraph (b) Clique expansion (c) Star expansion

- **Clique expansion:** If appropriately weighted, it has the same laplacian as the hypergraph: Spectrum-preserving graph coarsening can easily be applied. It is suitable for downsampling, but getting the associated hypergraph is NP-hard.

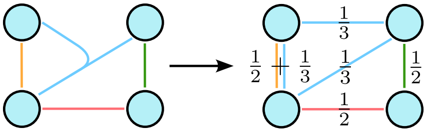
Why different projections?



(a) Original hypergraph (b) Clique expansion (c) Star expansion

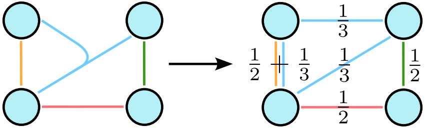
- **Clique expansion:** If appropriately weighted, it has the same laplacian as the hypergraph: Spectrum-preserving graph coarsening can easily be applied. It is suitable for downsampling, but getting the associated hypergraph is NP-hard.
- **Star expansion:** Suitable for the generation: Adding hyperedges and changing their content is easy (adding a right-side node and connecting left-side nodes to it). Getting the associated hypergraph is easy.

Coarsening sequence via weighted clique expansion



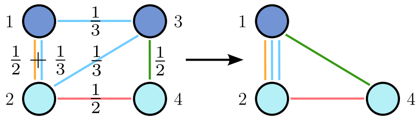
Each hyperedge is collapsed into a clique (*i.e.*, we connect the nodes in the hyperedge via regular edges) and we attribute the weight $1/|e|$ - where $|e|$ is the degree of the hyperedge - to these regular edges

Coarsening sequence via weighted clique expansion

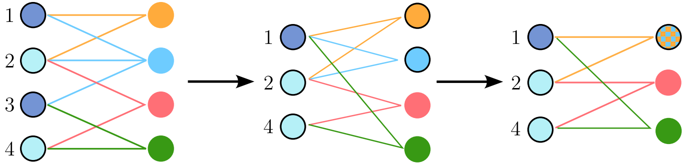


Each hyperedge is collapsed into a clique (*i.e.*, we connect the nodes in the hyperedge via regular edges) and we attribute the weight $1/|e|$ - where $|e|$ is the degree of the hyperedge - to these regular edges

- We select the nodes to merge via the standard procedure for graphs applied to the weighted clique expansion
- We maintain a parallel bipartite view of the hypergraph
- The final level of coarsening for the bipartite representation is two linked nodes with one on each side



(a) A coarsening step on the weighted clique expansion. Weights are omitted because they are important only for the initialization of the procedure.



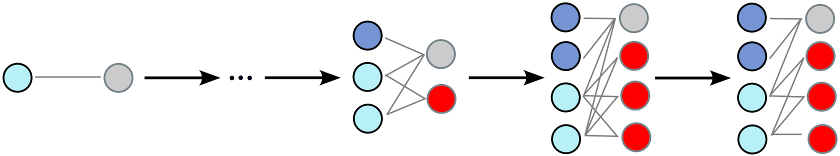
(b) Update of the equivalent bipartite representation.

We first update the left side of the bipartite representation then the right side by merging the nodes representing the same hyperedge.

Some remarks

- We have the following result: for a single merging of a pair of nodes on the left side, each merging of nodes on the right side involves at most three nodes.
- Only true for a single merging, but at each coarsening step there can be many. If the mergings are not carefully chosen, a chain reaction can happen where tens of hyperedges can merge at once.
- To circumvent this, in practice we discard the mergings resulting in more than 3 hyperedge fusions.

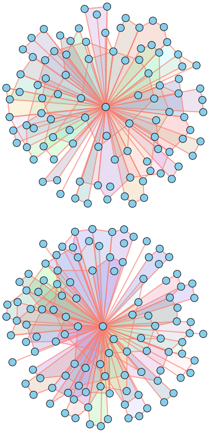
Generation procedure



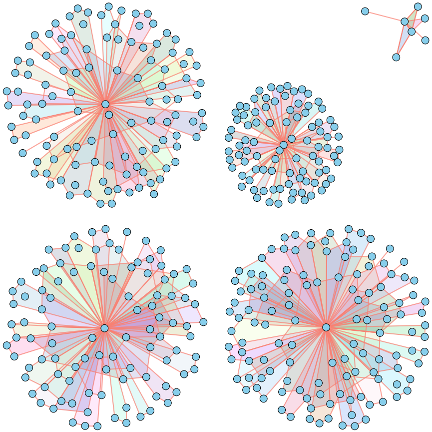
- Analogous to the case of the simple graph applied to the bipartite representation:
  1. Select nodes on the left to duplicate (up to 2 times)
  2. Select nodes on the right to duplicate (up to 3 times)
  3. Choose which edges to keep in the bipartite graph

TELECOM Paris IP PARIS

Some Visual Results (Ego Hypergraphs)

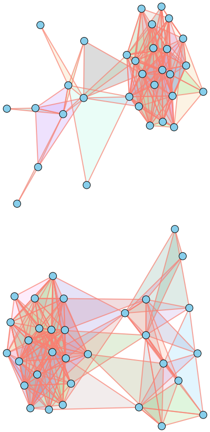


(a) Train samples

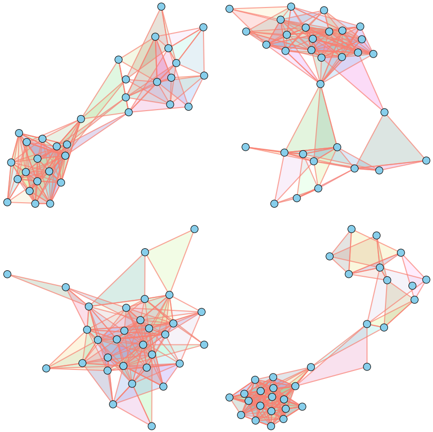


(b) Generated samples

Some Visual Results (Stochastic Block Model Hypergraphs)

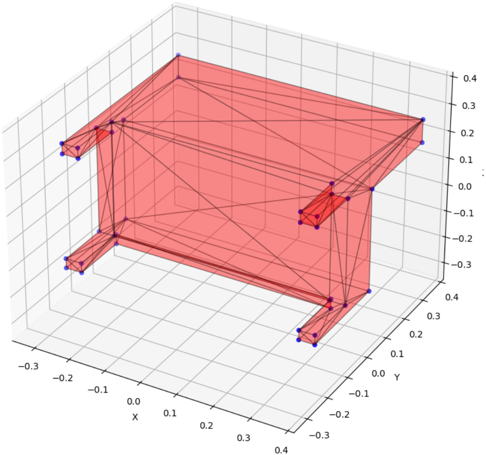


(a) Train samples



(b) Generated samples

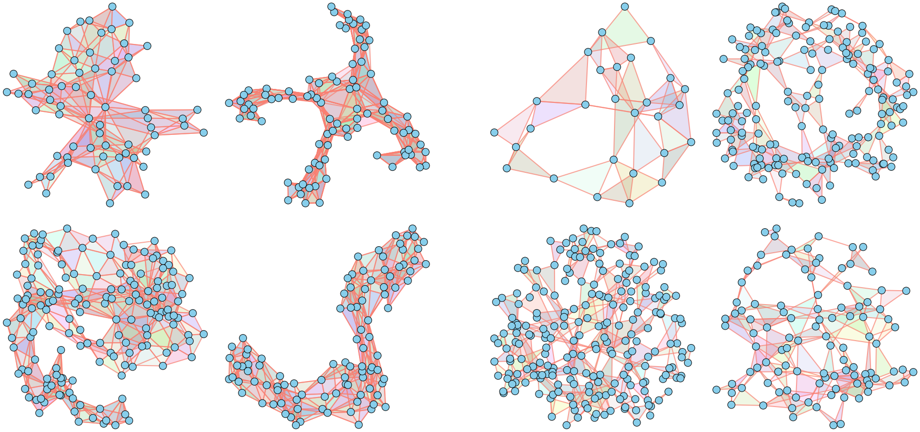
Real Data: Meshes



TELECOM Paris

IP PARIS

Some Visual Results (Meshes Topology)



(a) Train samples (b) Generated samples

Current Work

- **Limitation:** Our method generates featureless hypergraphs.
- **Idea:** follow the same paradigm of putting the information "locally" to take a "local decision".
  1. When merging nodes, merge their features (how? will depend on the manifold).
  2. When duplicating a node, both children get the feature of their parent.
  3. A model is trained to reconstruct the real features.

Main Takeaways and Conclusions

- Using classical graph methodologies for hypergraph generation is not straightforward

Main Takeaways and Conclusions

- Using classical graph methodologies for hypergraph generation is not straightforward
- Working in the bipartite representation (star expansion) seems like a promising avenue of research in hypergraph generation

Main Takeaways and Conclusions

- Using classical graph methodologies for hypergraph generation is not straightforward
- Working in the bipartite representation (star expansion) seems like a promising avenue of research in hypergraph generation
- The (weighted) clique expansion helps us use classical graph algorithms (like coarsening)

Main Takeaways and Conclusions

- Using classical graph methodologies for hypergraph generation is not straightforward
- Working in the bipartite representation (star expansion) seems like a promising avenue of research in hypergraph generation
- The (weighted) clique expansion helps us use classical graph algorithms (like coarsening)
- The number of nodes in the bipartite representation (nodes plus hyperedges) could increase very quickly, so efficient methodologies are required

Acknowledgments

Thank you!

Dorian Gailhard: dorian.gailhard@telecom-paris.fr
(https://doriangailhard.github.io/)

Appendices

Numerical Results

Model	SBM Hypergraphs ($n_{avg} = 31.73$, $std = 0.55$)					Ego Hypergraphs ($n_{avg} = 109.71$, $std = 10.23$)					Tree Hypergraphs ($n_{avg} = 32$, $std = 0$)				
	Valid SBM ↑	**Node Num ↓**	**Node Deg ↓**	**Edge Size ↓**	**Spectral ↓**	**Valid Ego ↑**	**Node Num ↓**	**Node Deg ↓**	**Edge Size ↓**	**Spectral ↓**	**Valid Tree ↑**	**Node Num ↓**	**Node Deg ↓**	**Edge Size ↓**	**Spectral ↓**
HyperPA	2.5%	**0.075**	4.062	0.407	0.273	0%	35.83	2.590	0.423	0.237	0%	2.350	0.315	0.284	0.159
VAE	0%	0.375	1.280	1.059	0.024	0%	47.58	0.803	1.458	0.133	0%	9.700	0.072	0.480	0.124
GAN	0%	1.200	2.106	1.203	0.059	0%	60.35	0.917	1.665	0.230	0%	6.000	0.151	0.469	0.089
Diffusion	0%	0.150	1.717	1.390	0.031	0%	**4.475**	3.984	2.985	0.190	0%	2.225	1.718	1.922	0.127
HYGENE	**65%**	0.525	**0.321**	**0.002**	**0.010**	**90%**	12.55	**0.063**	**0.220**	**0.004**	**77.5%**	**0.000**	**0.059**	**0.108**	**0.012**

Model	Erdos-Renyi Hypergraphs ($n_{avg} = 32$, $std = 0.07$)				ModelNet40 Piano ($n_{avg} = 177.29$, $std = 57.11$)				ModelNet40 Plant ($n_{avg} = 124.86$, $std = 87.88$)				ModelNet40 Bookshelf ($n_{avg} = 119.38$, $std = 68.20$)			
	Node Num ↓	**Node Deg ↓**	**Edge Size ↓**	**Spectral ↓**	**Node Num ↓**	**Node Deg ↓**	**Edge Size ↓**	**Spectral ↓**	**Node Num ↓**	**Node Deg ↓**	**Edge Size ↓**	**Spectral ↓**	**Node Num ↓**	**Node Deg ↓**	**Edge Size ↓**	**Spectral ↓**
HyperPA	**0.000**	5.530	0.183	0.177	0.825	9.254	**0.023**	**0.067**	10.83	6.566	0.046	0.061	8.025	7.562	0.044	**0.048**
VAE	0.100	2.140	0.540	0.035	75.35	8.060	1.686	0.396	76.15	3.895	1.573	0.205	47.45	6.190	1.520	0.190
GAN	0.675	2.560	0.657	0.048	**0.000**	409.0	86.38	0.697	**0.000**	378.1	56.35	0.364	**0.000**	397.2	46.30	0.476
Diffusion	0.050	2.225	0.781	0.014	0.050	20.90	4.192	0.113	0.025	21.03	3.439	0.069	**0.000**	20.36	2.346	0.079
HYGENE	0.775	**0.445**	**0.012**	**0.006**	42.52	**6.290**	0.027	0.117	68.38	**2.428**	**0.027**	**0.034**	69.73	**1.050**	**0.034**	0.068