Molecular Hypergraph Grammar with Its Application to Molecular Optimization

Hiroshi Kajino

MIT-IBM Watson AI Lab
IBM Research - Tokyo

We wish to learn a generative model of a molecule.

- Generative model of a molecule $p(G \mid \boldsymbol{z})$ [Gómez-Bombarelli+, 16]
  - Input: Latent vector $\boldsymbol{z} \in \mathbb{R}^D \sim \mathcal{N}(0, I_D)$
  - Output: Molecular graph G (graph w/ node labels)

Continuous optimization problem ⇔ Molecular optimization problem

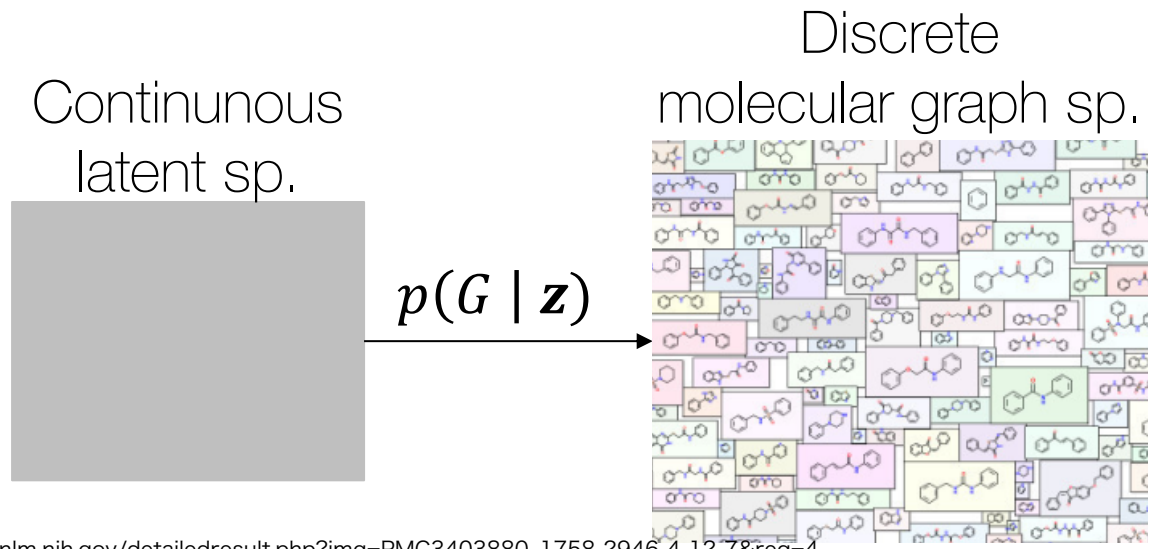
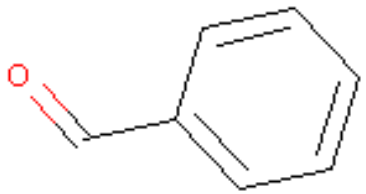


Image from https://openi.nlm.nih.gov/detailedresult.php?img=PMC3403880_1758-2946-4-12-7&req=4

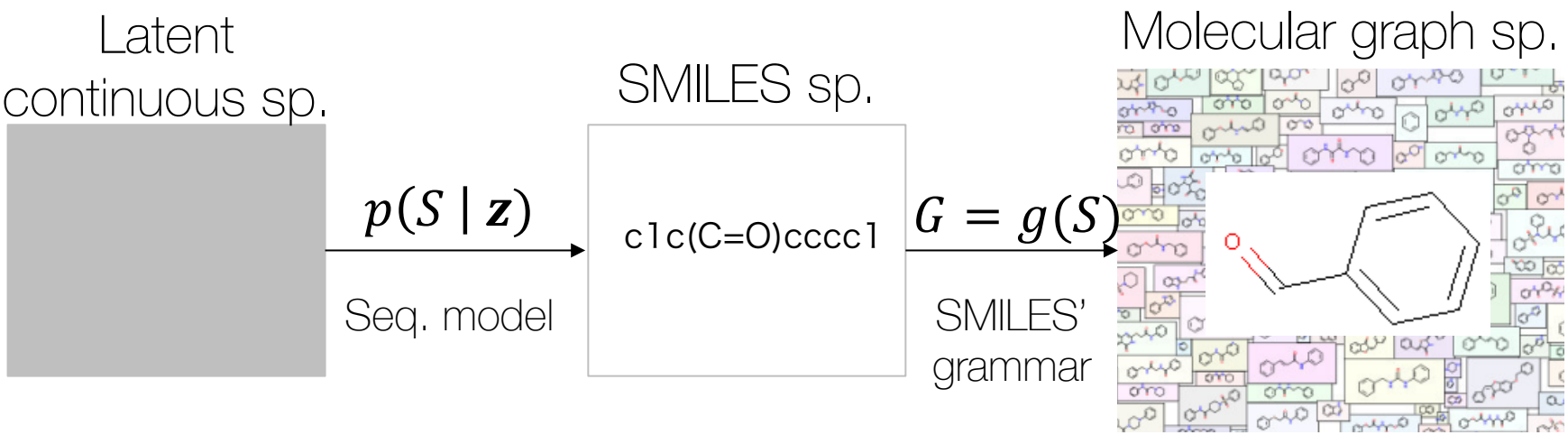
Molecular graph generation is a non-trivial task

- Two technical challenges
  1. No consensus on a generative model of a graph
     - LSTM, GRU for path and tree
     - Ring is non-trivial
  2. Hard constraints such as valence conditions
     - Degree of each node (=atom) is specified by its label
     - e.g., carbon has degree 4, oxygen has degree 2.



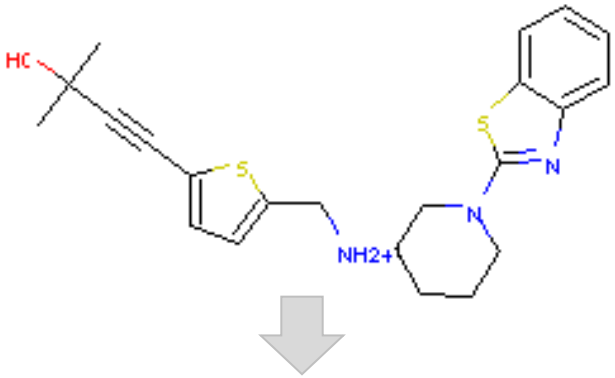
Most of the existing work employs a text representation of a molecule, called SMILES.

- Use text representation "SMILES" [Gómez-Bombarelli+, 16]
  - –Pros: a standard sequential model can generate SMILES



Statistical model has to learn a rule-based grammar

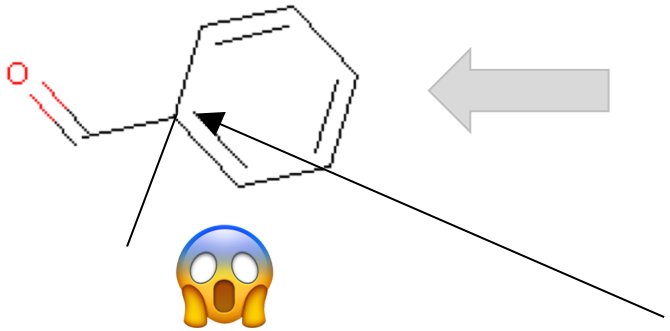
- Use text representation "SMILES" [Gómez-Bombarelli+, 16]
  - –Pros: a standard sequential model can generate SMILES
  - –Cons:
    - NN has to learn SMILES' grammar
    - No guarantee on valence conditions



CC(C)(O)C#Cc1ccc(C[NH2+][C@H]2CCCN(c3nc4ccccc4s3)C2)s1 😱

SMILES' grammar does not prescribe valence conditions

- Use text representation "SMILES" [Gómez-Bombarelli+, 16]
  - –Pros: a standard sequential model can generate SMILES
  - –Cons:
    - NN has to learn SMILES' grammar
    - No guarantee on valence conditions



Grammatically correct

c1c(C=O)(C)cccc1

This atom has a valence of 5.
That of carbon must be 4.

JT-VAE achieves 100% validity for the first time, but requires multiple NNs.

- Generate a molecule by assembling subgraphs [Jin+, 18]

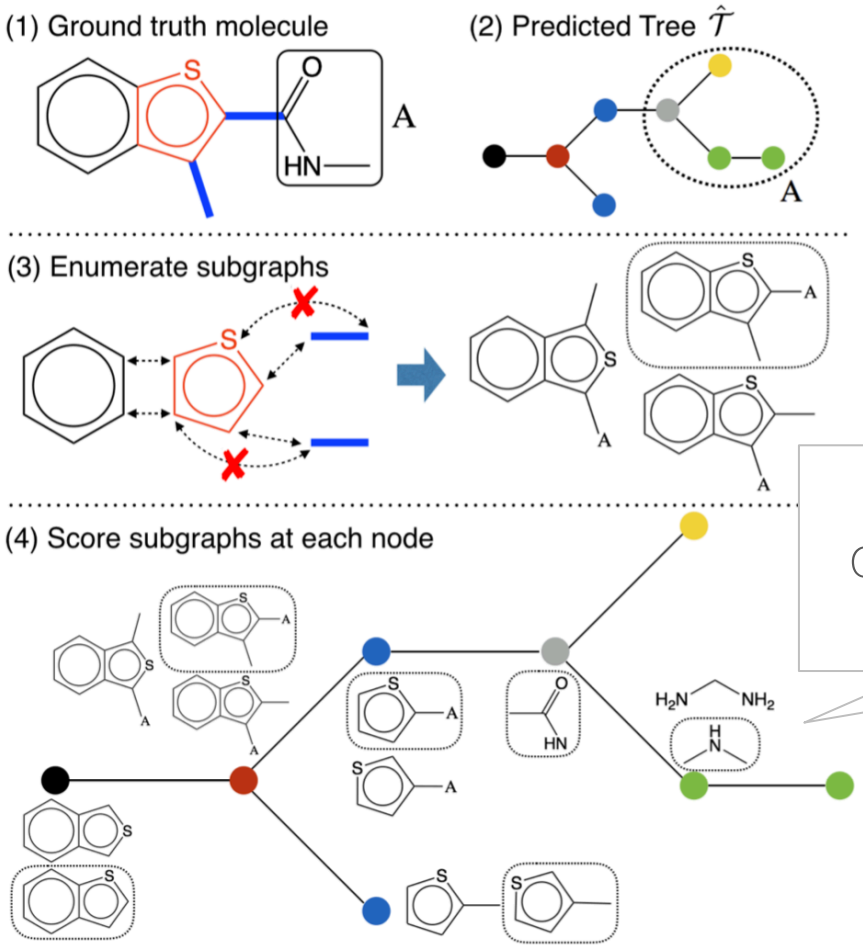
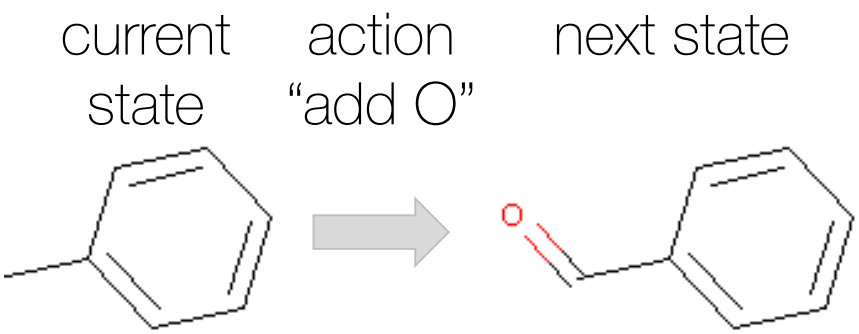


Image from [Jin+, 18]

RL-based method discovers better molecules than VAE-based methods, but with enormous cost

- Molecular optimization using RL [You+, 18]
  - Idea: Molecular generation as MDP
    - State: Molecular graph
    - Action: Modify the graph
    - Reward: Target property



  - Pros: Better optimization capability than VAE-based methods
  - Cons: Requires a number of target property evaluations
    - # of evals $> 10^9$ 😱
    - Infeasible to work with the first principle calculation / wet-lab experiments

No existing work satisfies all of the three properties

	VAE-based		RL-based
	SMILES	Graph	Graph
Validity		✓	✓
Easy-to-generate	✓		
Sample complexity	✓	✓	

Our contribution is to facilitate graph-based generation with help of "graph grammar".

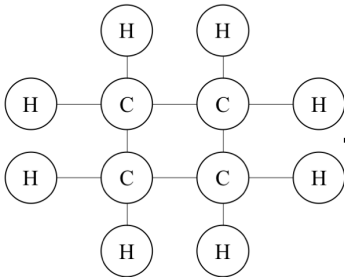
	VAE-based		RL-based
	SMILES	Graph	Graph
Validity		✓	✓
Easy-to-generate	✓	✓	✓?
Sample complexity	✓	✓	

We develop a graph grammar tailored for molecular generation

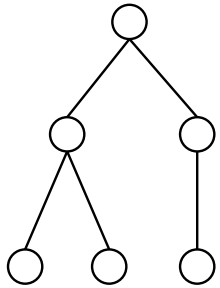
- Idea

Use a context-free graph grammar for graph generation

– Graph generation boils down to *tree generation*



=



Easy to generate 🥳

– Requirements:

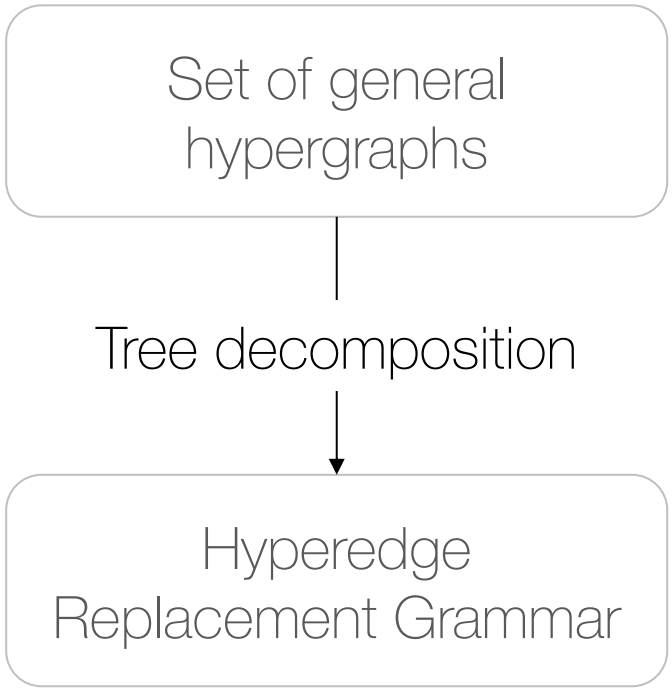
- Always satisfy valence conditions ← Our main contribution
- Context-freeness
- Inference algorithm from data, not hand-written rules

} Hyperedge replacement grammar

A hyperedge replacement grammar can be constructed from data via tree decomposition

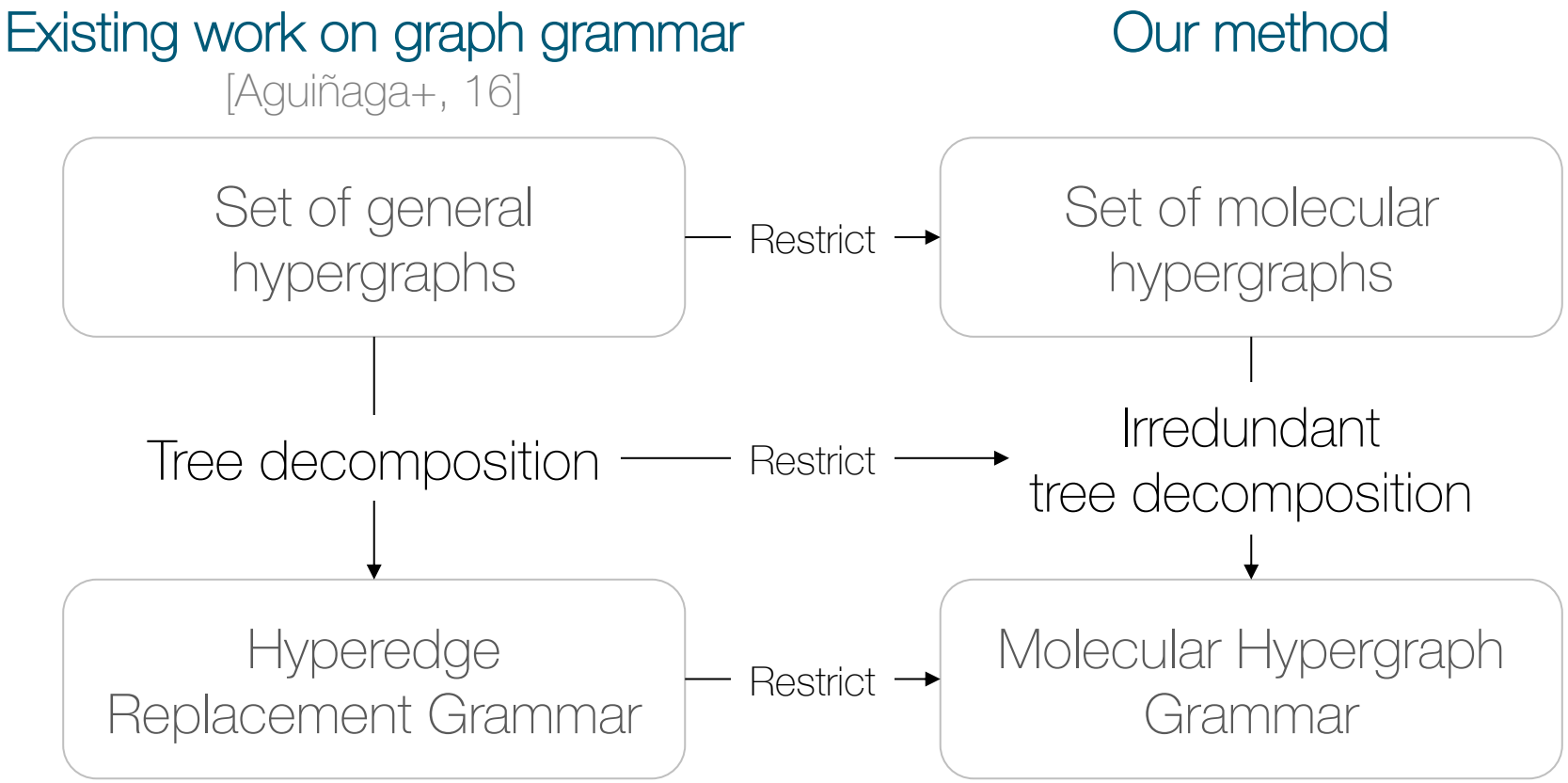
Existing work on graph grammar

[Aguiñaga+, 16]



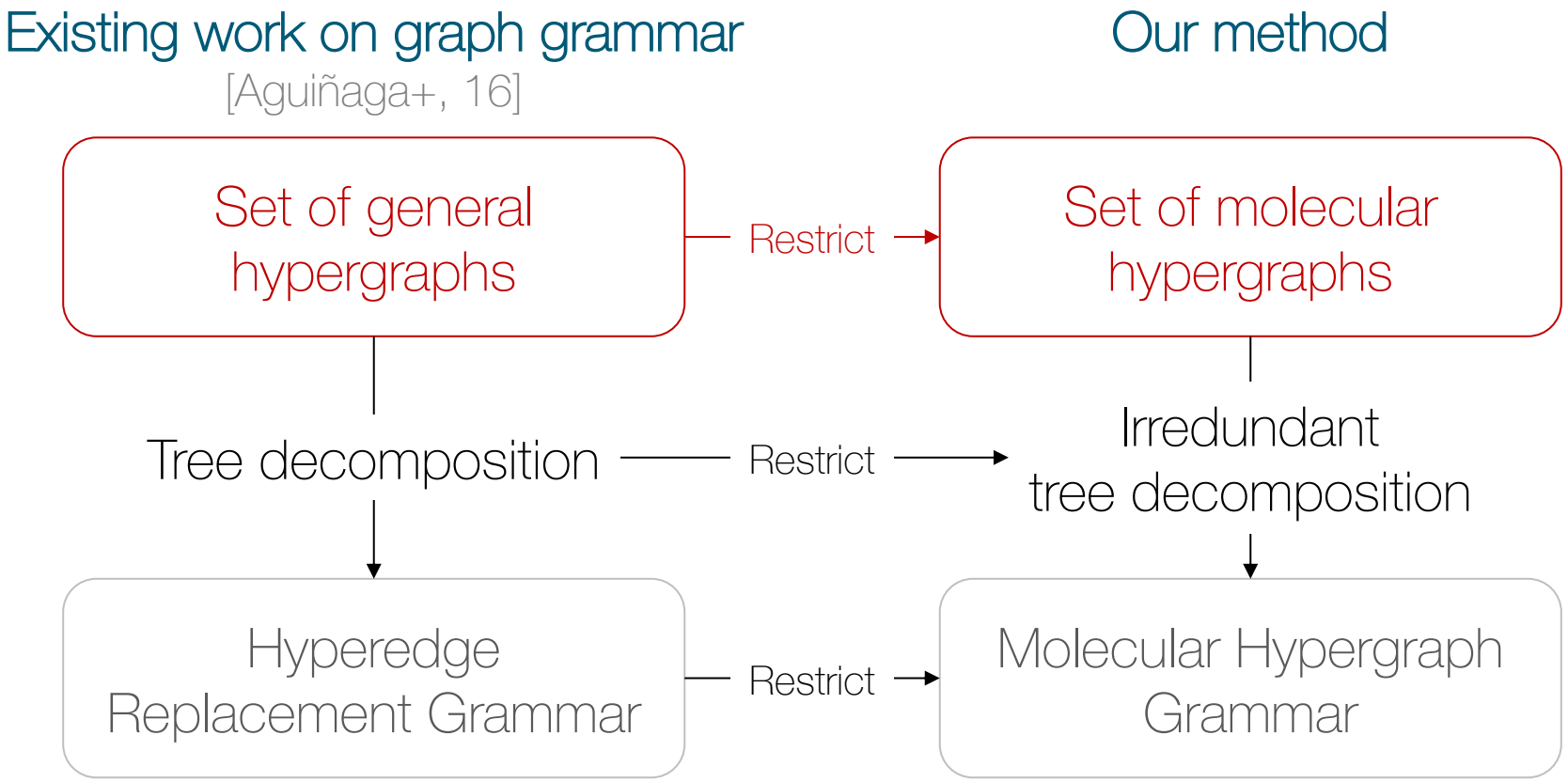
* HRG is a context-free grammar generating hypergraphs

Each restriction is our contribution in the literature of graph grammar



* HRG is a context-free grammar generating hypergraphs

Each restriction is our contribution in the literature of graph grammar

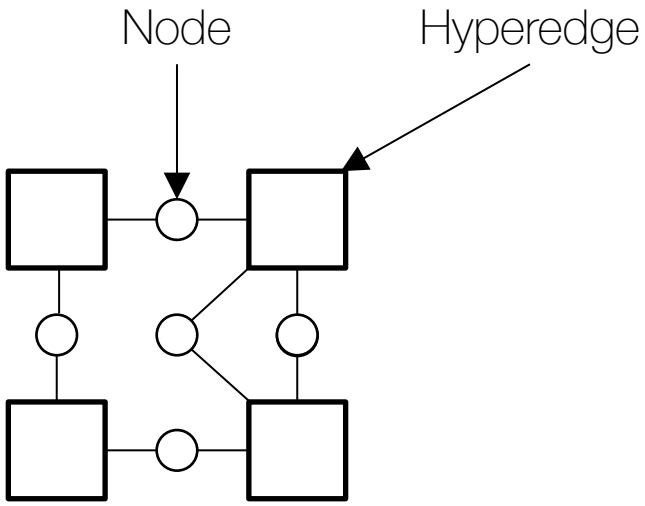


* HRG is a context-free grammar generating hypergraphs

Hypergraph is a generalization of a graph

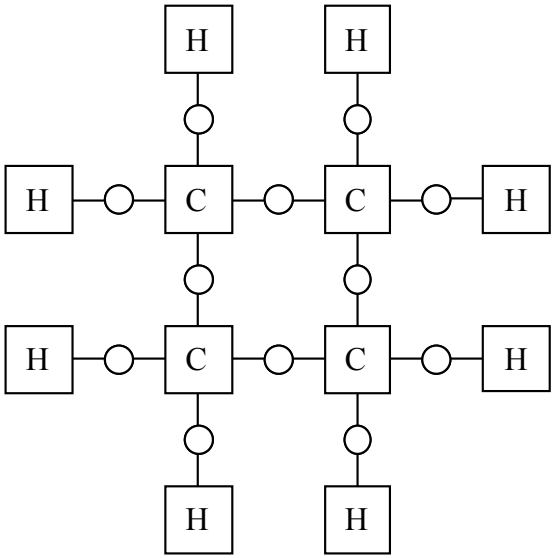
- Hypergraph $\mathcal{H} = (V, E)$ consists of…
  - Node $v \in V$
  - Hyperedge $e \in E \subseteq 2^{|V|}$: Connect an arbitrary number of nodes

    cf, An edge in a graph connects exactly two nodes

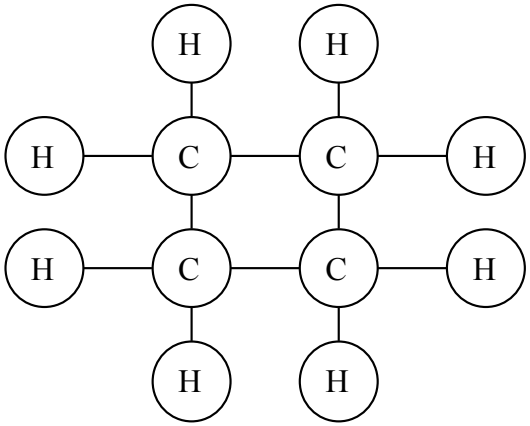


We represent a molecule using a hypergraph, not a graph. This helps to satisfy the valence conditions.

- Molecular hypergraph models…
  - Atom = hyperedge
  - bond = node

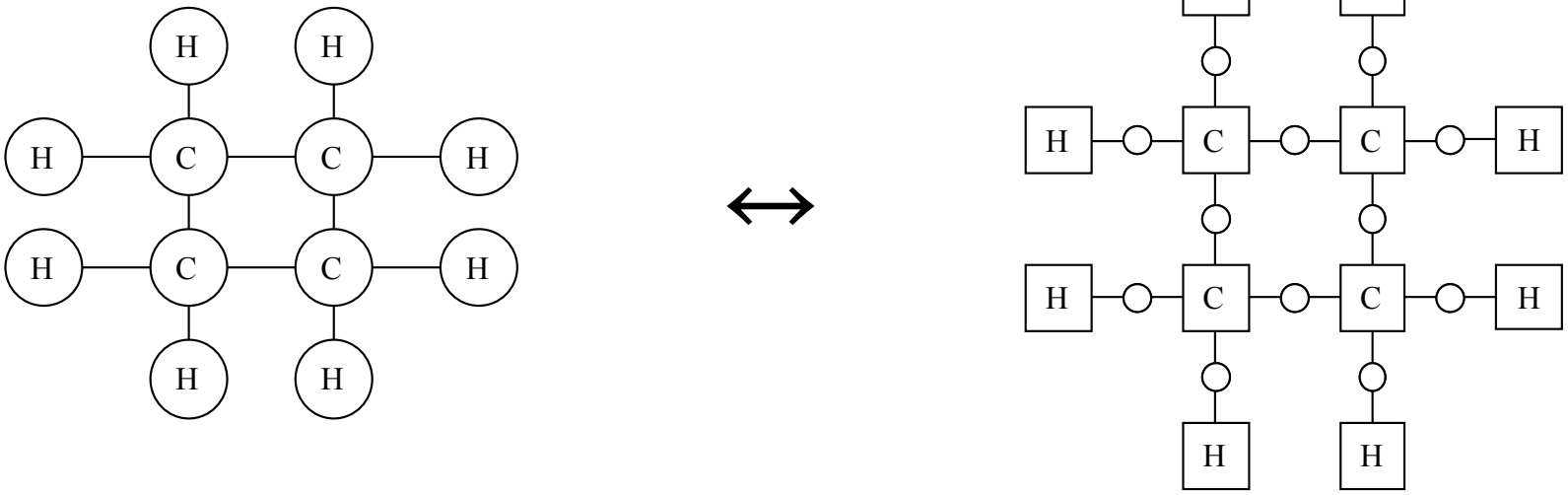


- Molecular graph models…
  - Atom = node
  - Bond = Edge

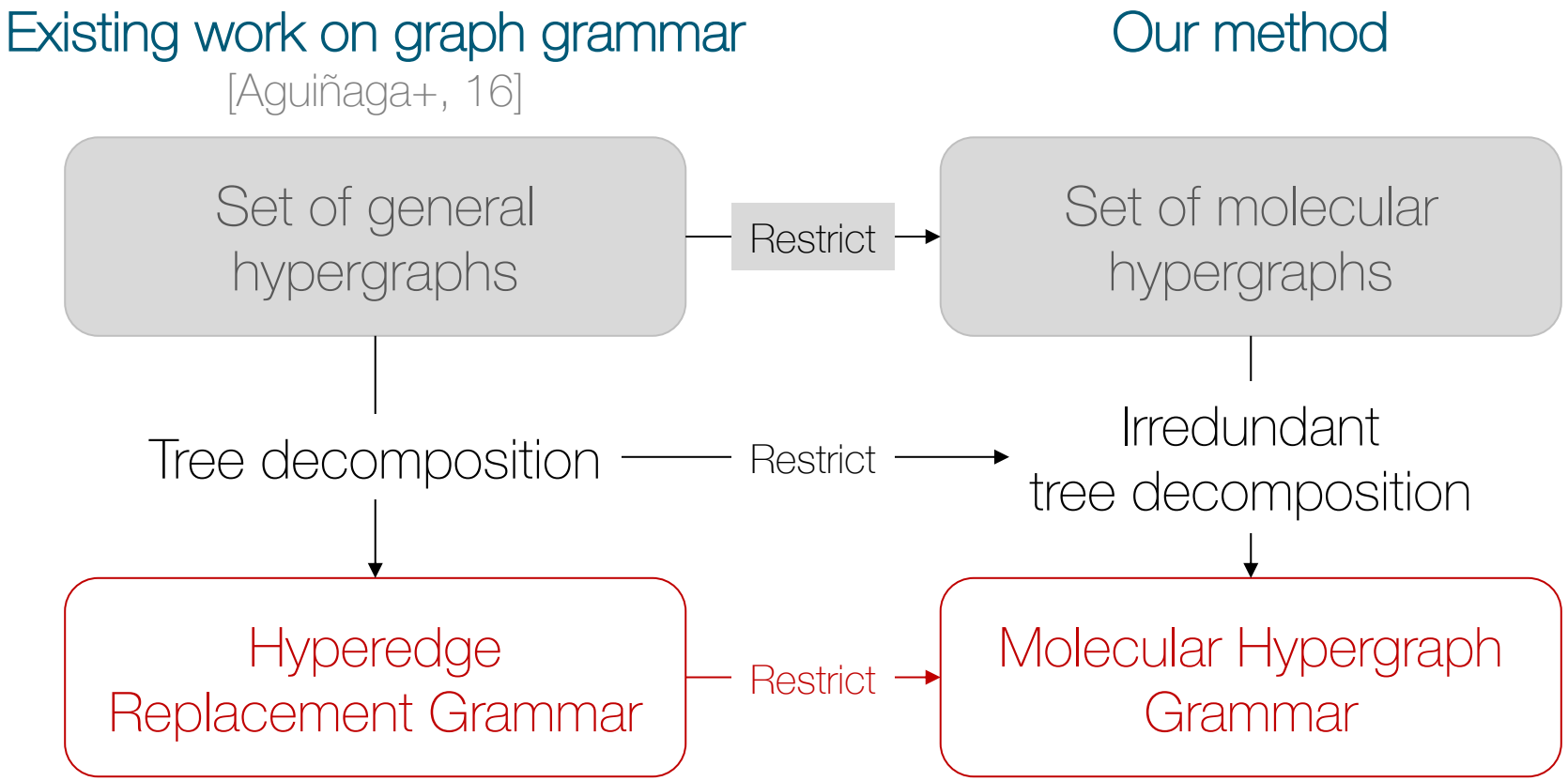


These requirements guarantee transformation between graph & hypergraph

- Molecular hypergraph
  1. Each node has degree 2 (=2-regular)
  2. Label on a hyperedge determines # of nodes it has (= valence)



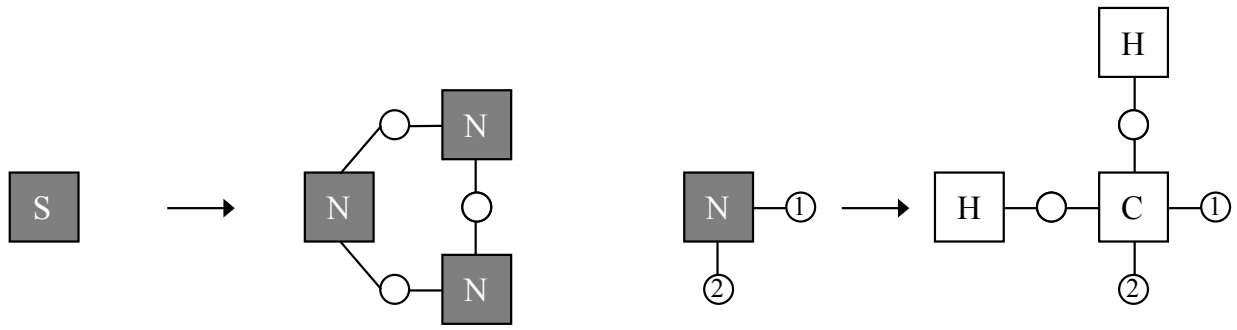
Each restriction is our contribution in the literature of graph grammar



* HRG is a context-free grammar generating hypergraphs

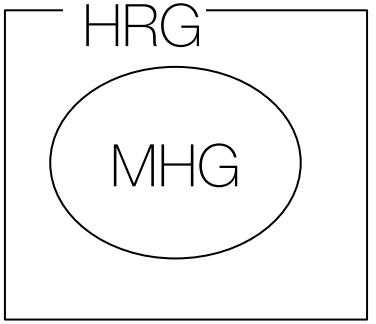
HRG generates a hypergraph by repeatedly replacing non-terminal hyperedges with hypergraphs

- Hyperedge replacement grammar (HRG) $\mathcal{G} = (N, T, S, P)$
  - N: set of non-terminals
  - T: set of terminals
  - S: starting symbol
  - P: set of production rules
    - A rule replaces a non-terminal hyperedge with a hypergraph



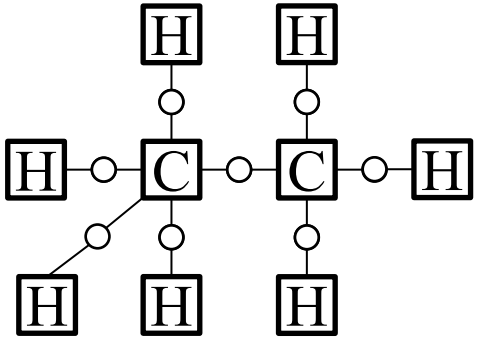
Replace l.h.s. with r.h.s.

MHG is defined as a subclass of HRG

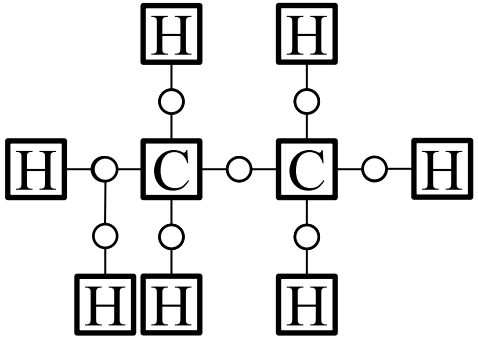


- Molecular Hypergraph Grammar (MHG)
  - –**Definition:** HRG that generates molecular hypergraphs only
  - –**Counterexamples:**

Valence 😡

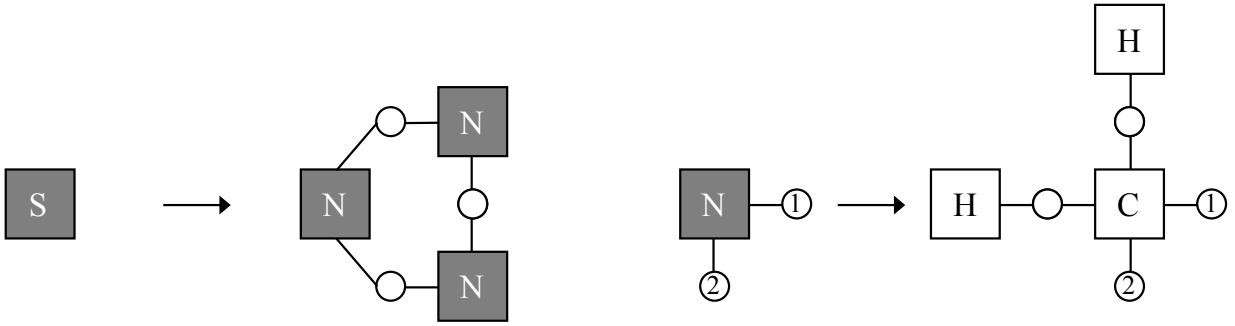


2-regularity 😡



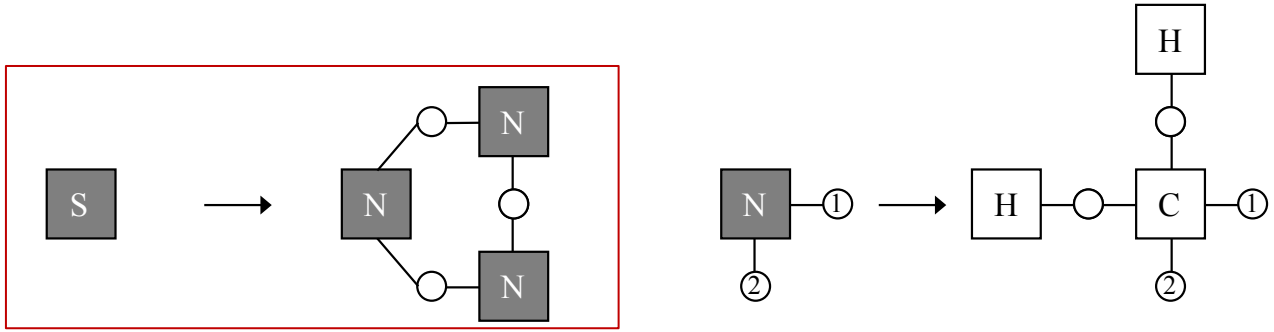
Start from starting symbol **S**

Production rules P

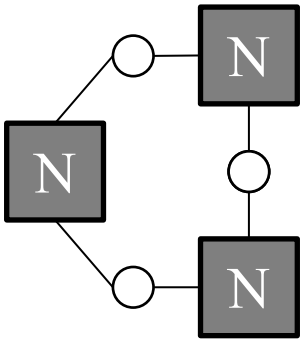


The left rule is applicable

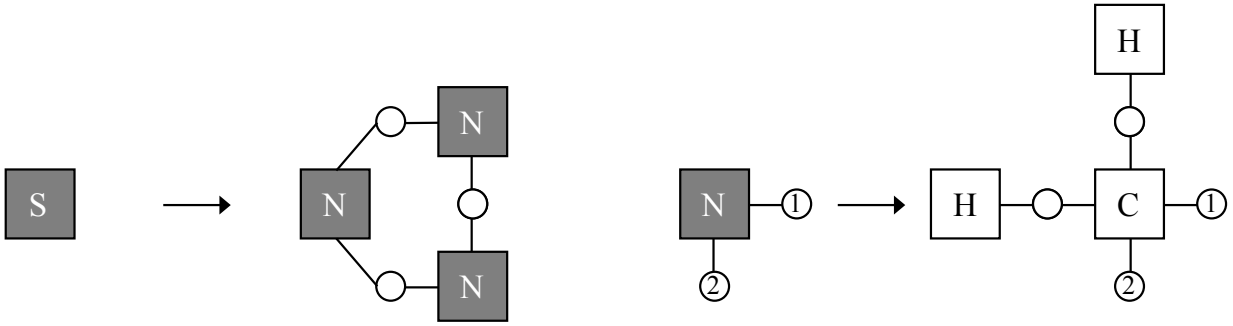
Production rules P



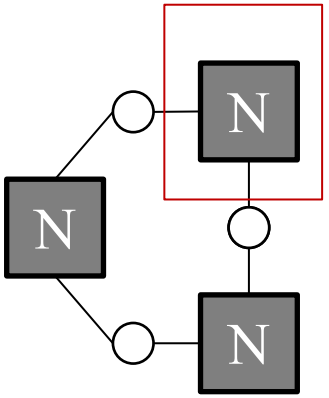
We obtain a hypergraph with three non-terminals



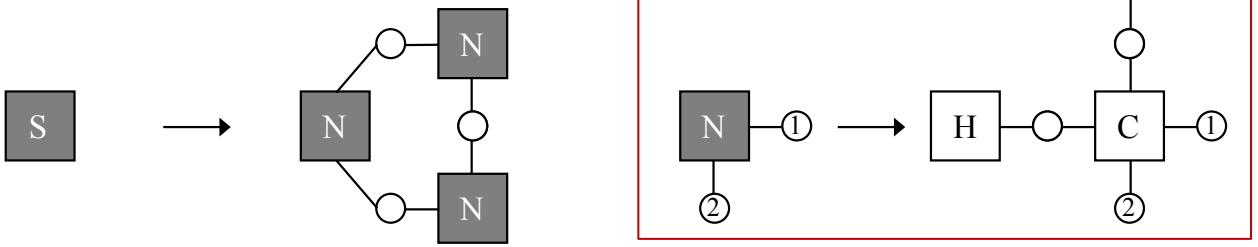
Production rules P



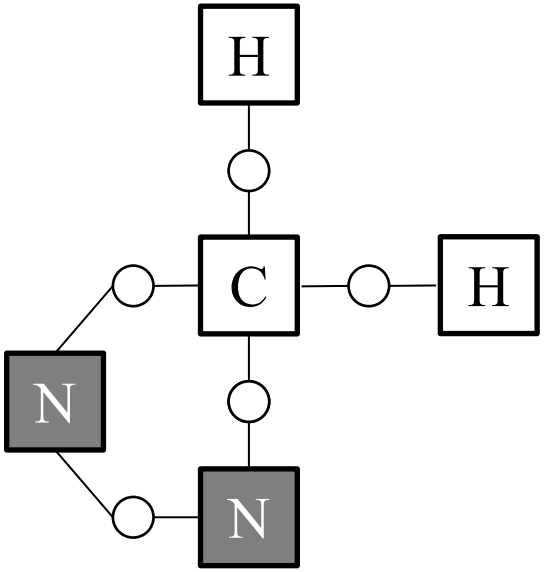
Apply the right rule to one of the non-terminals



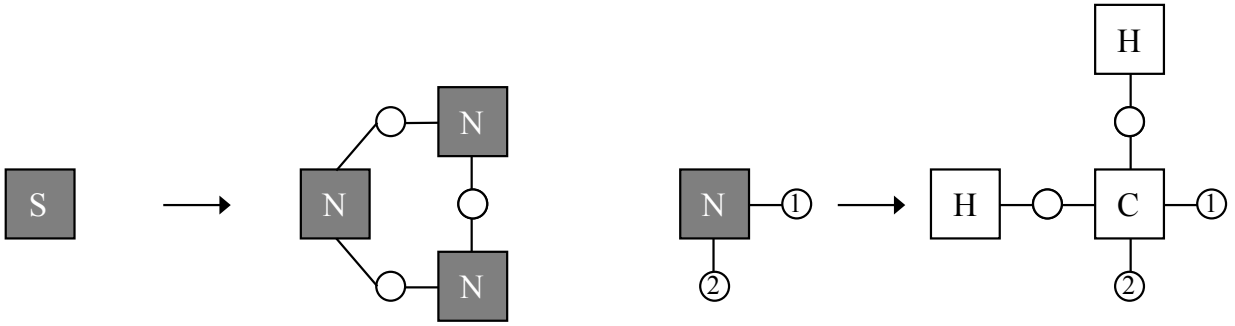
Production rules P



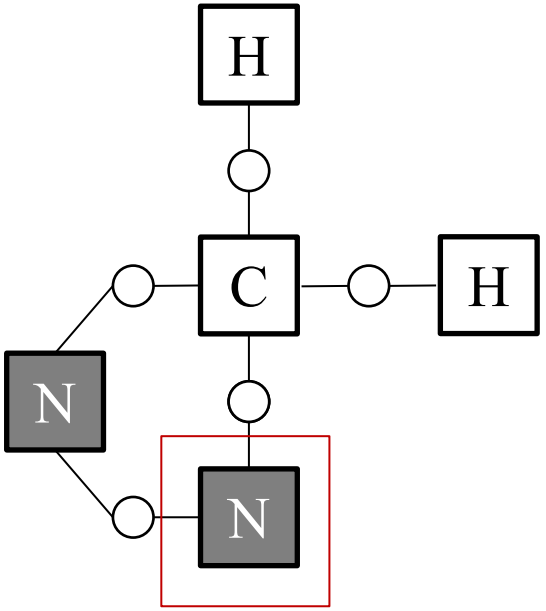
Two non-terminals remain



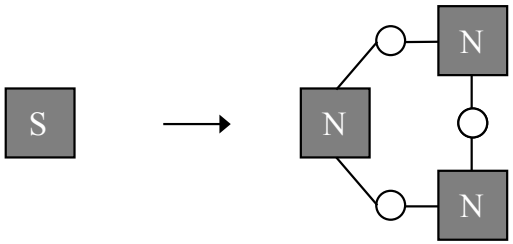
Production rules P

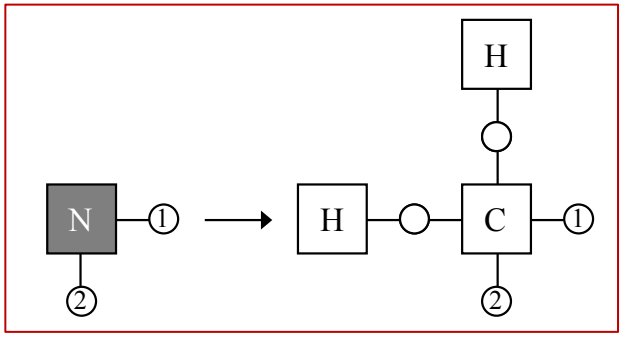


Repeat the procedure until there is no non-terminal

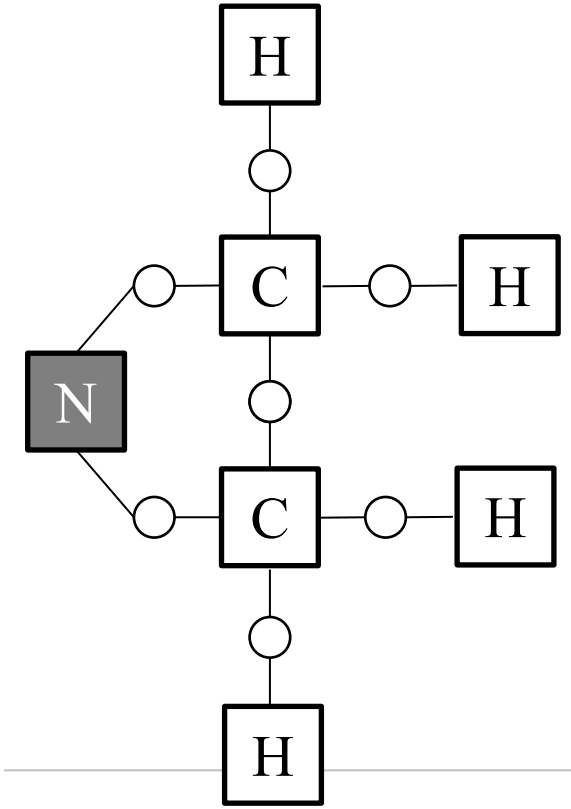


Production rules P

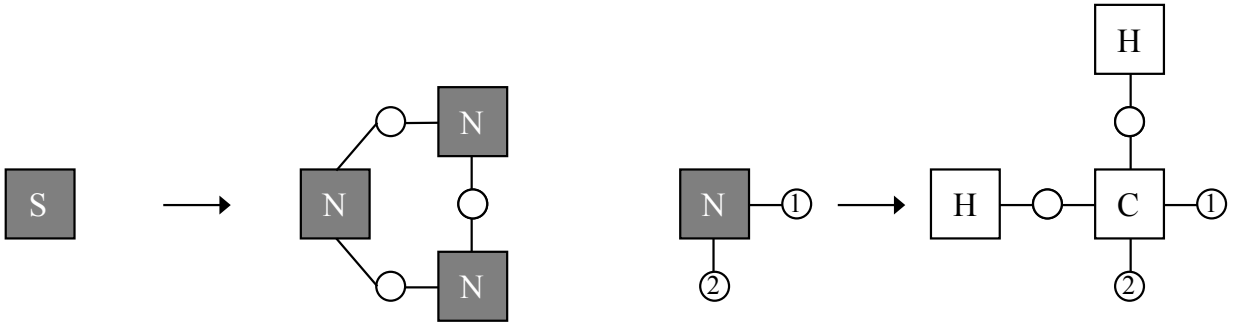




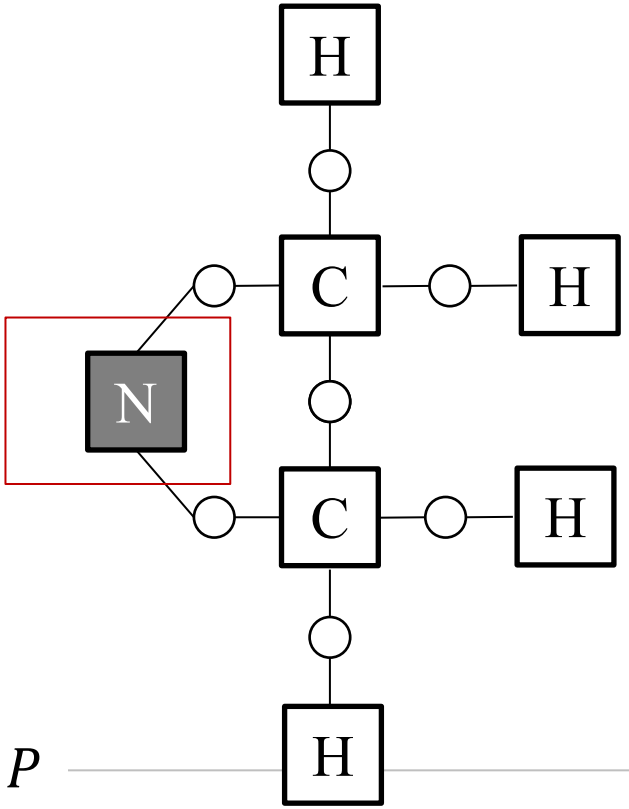
Repeat the procedure until there is no non-terminal



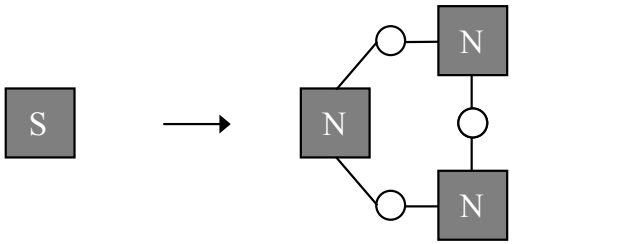
Production rules P

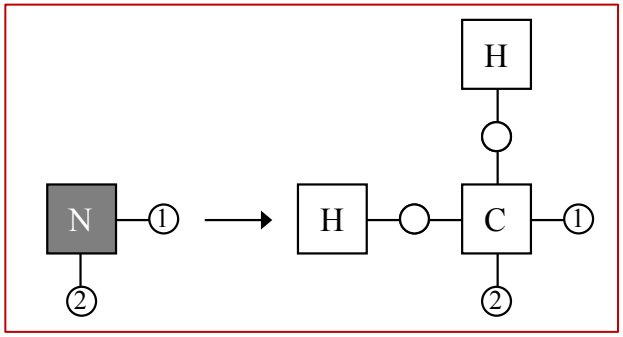


Repeat the procedure until there is no non-terminal

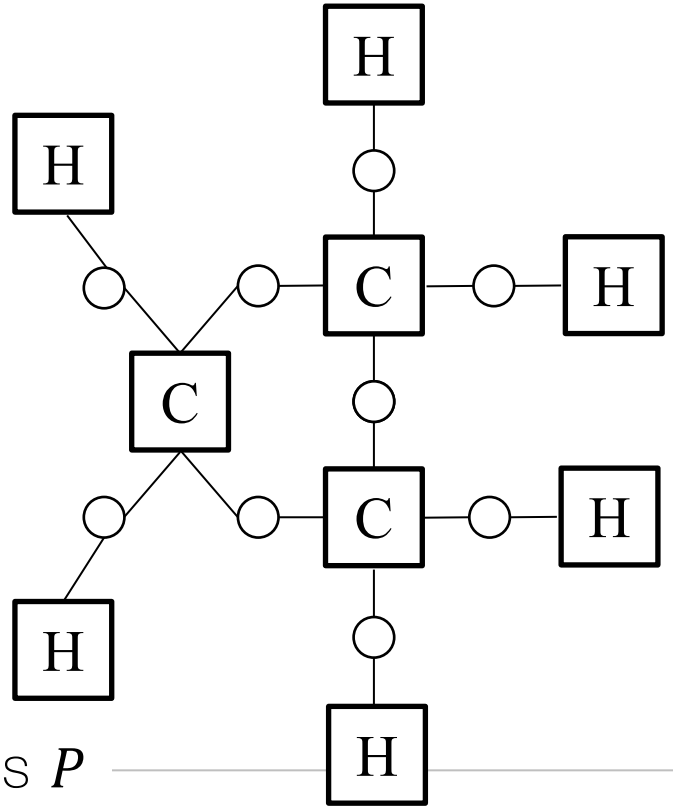


Production rules P

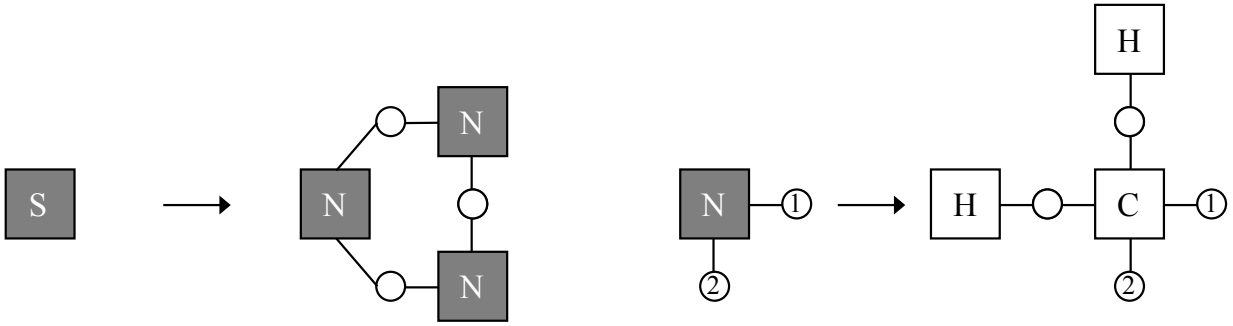




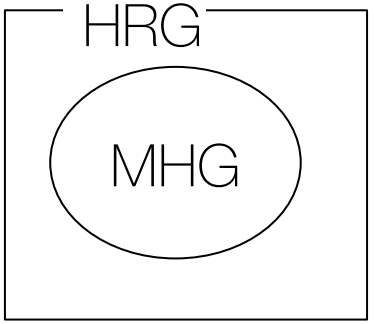
Graph generation halts when there is no non-terminal,



Production rules P

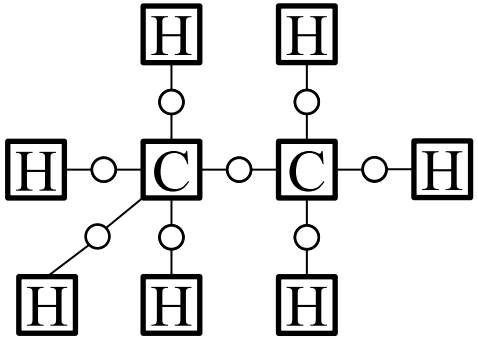


MHG is defined as a subclass of HRG

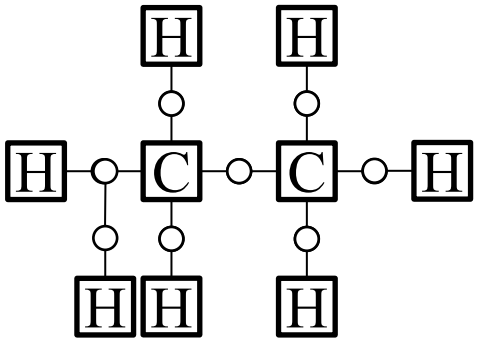


- Molecular Hypergraph Grammar (MHG)
  - **Definition:** HRG that generates molecular hypergraphs only
  - **Counterexamples:**

Valence 😡

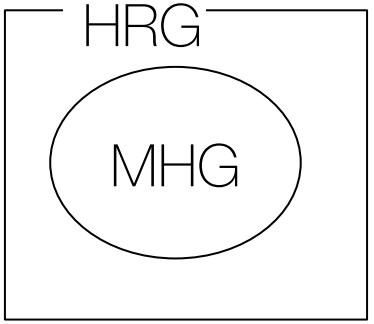


2-regularity 😡



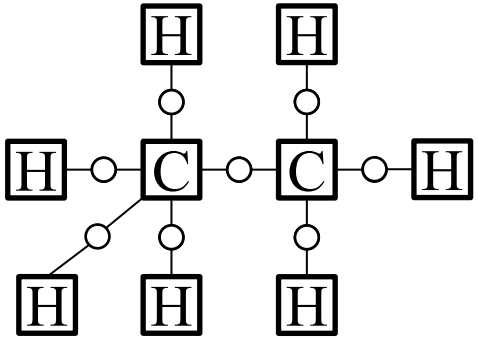
This can be avoided by learning HRG from data [Aguiñaga+, 16]

MHG is defined as a subclass of HRG



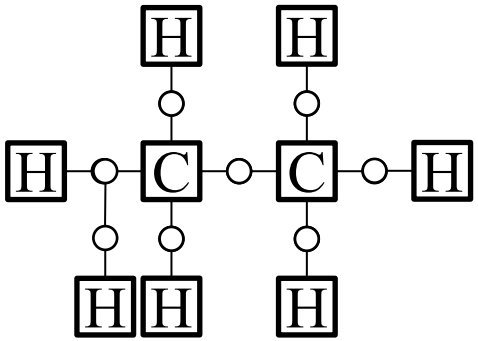
- Molecular Hypergraph Grammar (MHG)
  - –**Definition:** HRG that generates molecular hypergraphs only
  - –**Counterexamples:**

Valence 😡



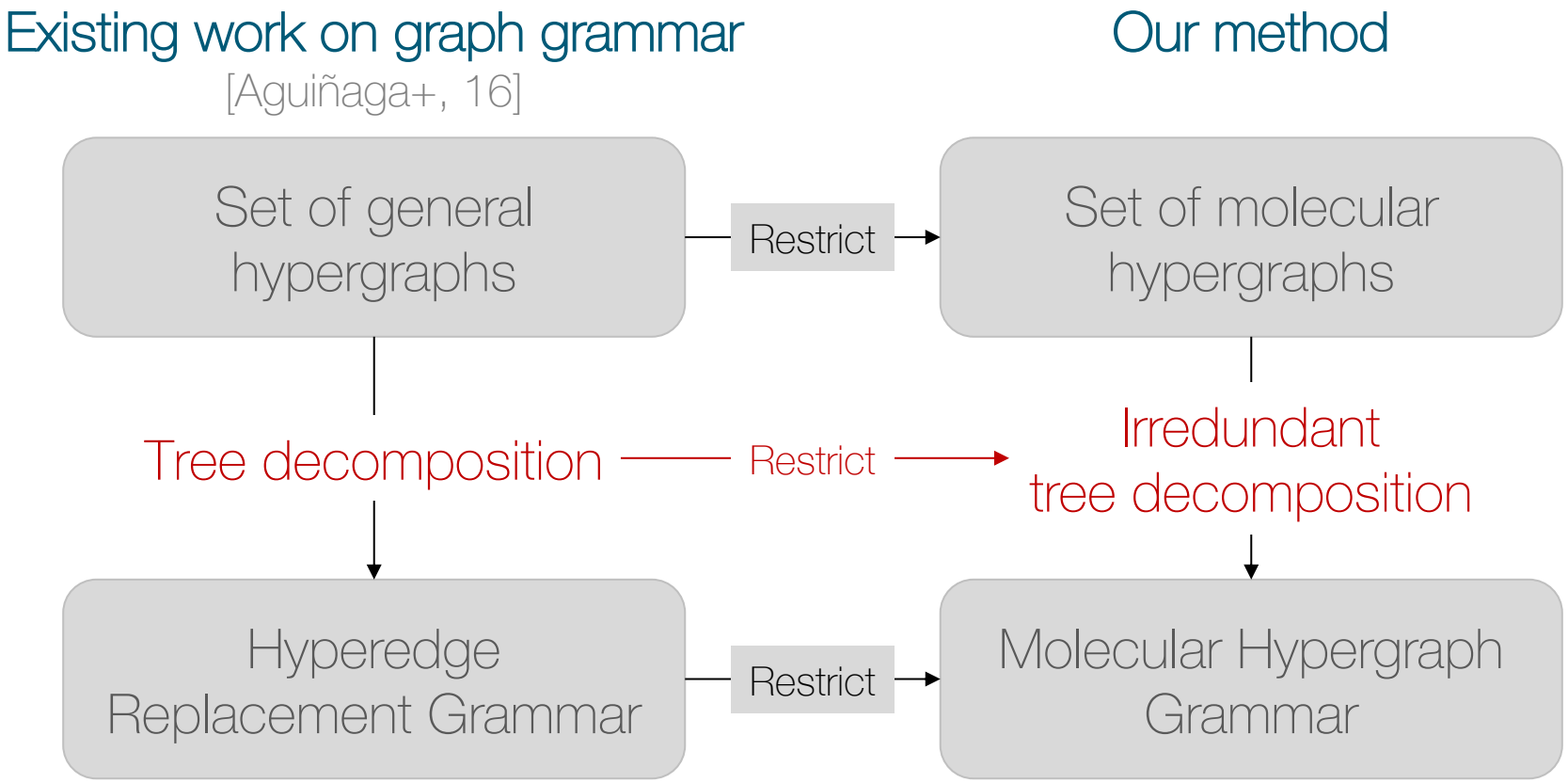
This can be avoided by learning HRG from data [Aguiñaga+, 16]

2-regularity 😡



Use an irredundant tree decomposition (our contribution)

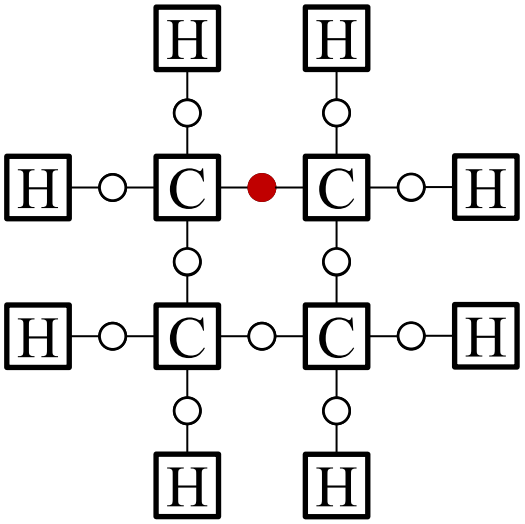
Each restriction is our contribution in the literature of graph grammar

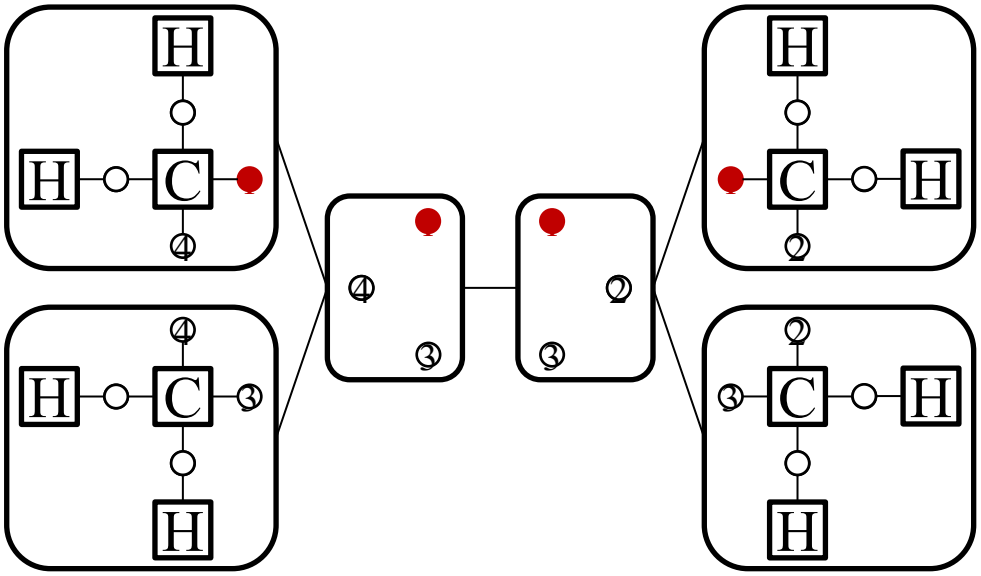


* HRG is a context-free grammar generating hypergraphs

Tree decomposition discovers a tree-like structure in a graph

- Tree decomposition
  - All of the nodes and edges must be included in the tree
  - For each node, the tree nodes that contain it must be connected

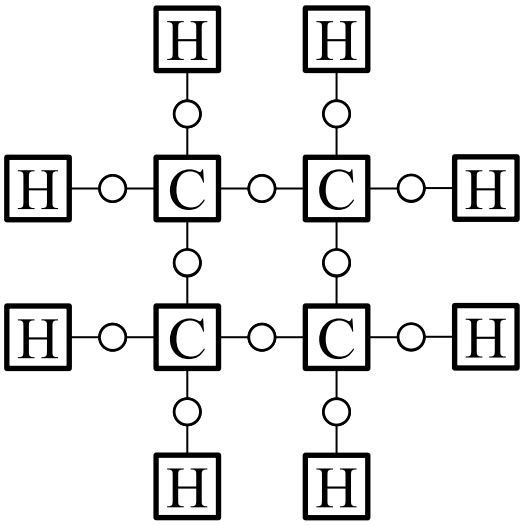


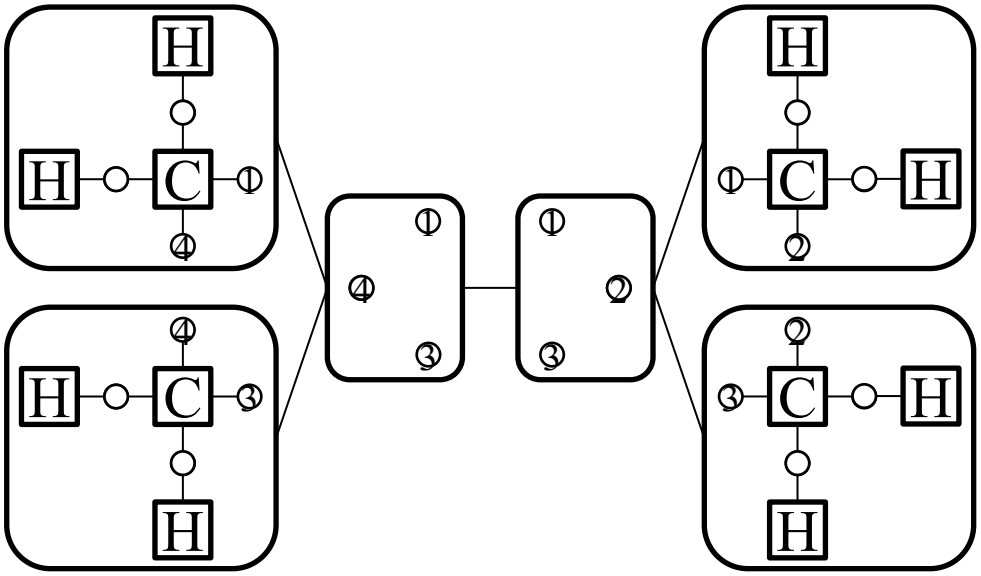


* Digits represent the node correspondence

Tree decomposition and (a syntax tree of) HRG are equivalent

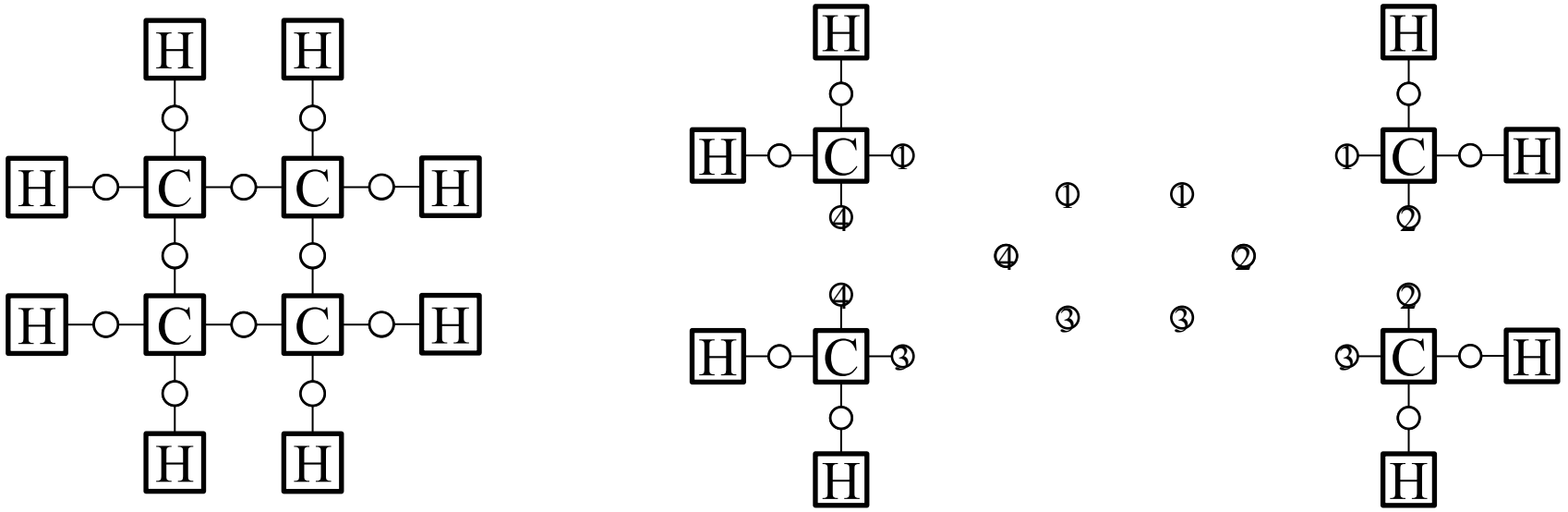
- Relationship between tree decomposition and HRG
  1. Connecting hypergraphs in tree recovers the original hypergraph
  2. Connection ⇔ Hyperedge replacement





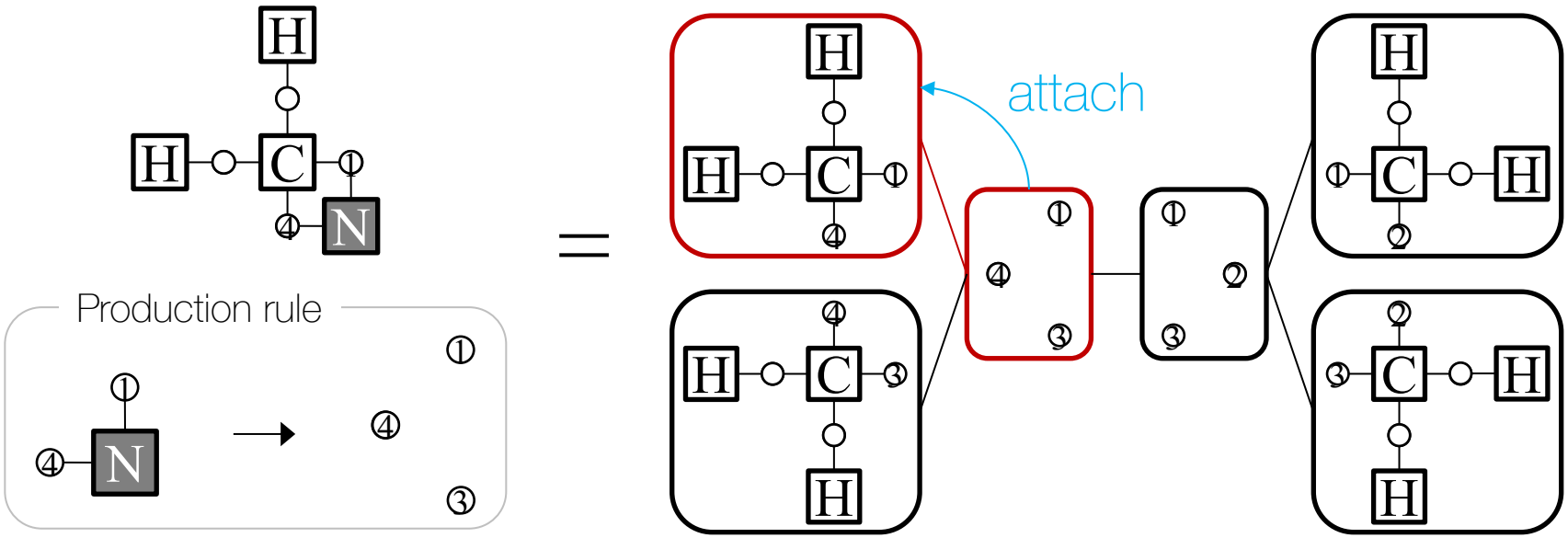
Tree decomposition and (a syntax tree of) HRG are equivalent

- Relationship between tree decomposition and HRG
  1. Connecting hypergraphs in tree recovers the original hypergraph
  2. Connection ⇔ Hyperedge replacement



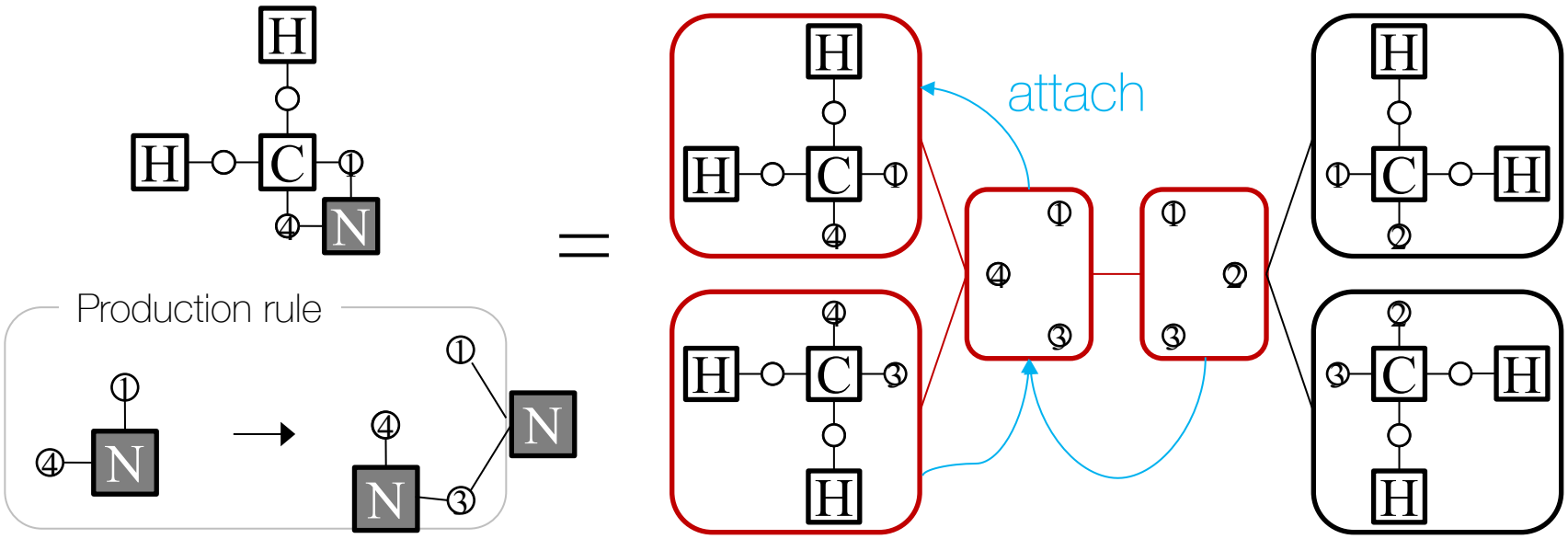
Tree decomposition and (a syntax tree of) HRG are equivalent

- Relationship between tree decomposition and HRG
  1. Connecting hypergraphs in tree recovers the original hypergraph
  2. Connection ⇔ Hyperedge replacement



Tree decomposition and (a syntax tree of) HRG are equivalent

- Relationship between tree decomposition and HRG
  1. Connecting hypergraphs in tree recovers the original hypergraph
  2. Connection ⇔ Hyperedge replacement



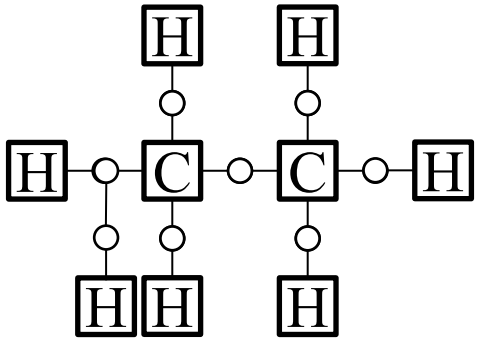
Extract production rules from a tree decomposition; then HRG with the rules can reconstruct the original hypergraph

- HRG inference algorithm [Aguiñaga+, 16]
  - **Input:** Set of hypergraphs
  - **Output:** HRG w/ the following properties:
    - All of the input hypergraphs are in the language 😀
    - Guarantee the valence conditions 😀
    - No guarantee on 2-regularity 😭

1. Compute tree decompositions of input hypergraphs
2. Extract production rules
3. Compose HRG by taking their union

Extract production rules from a tree decomposition; then HRG with the rules can reconstruct the original hypergraph

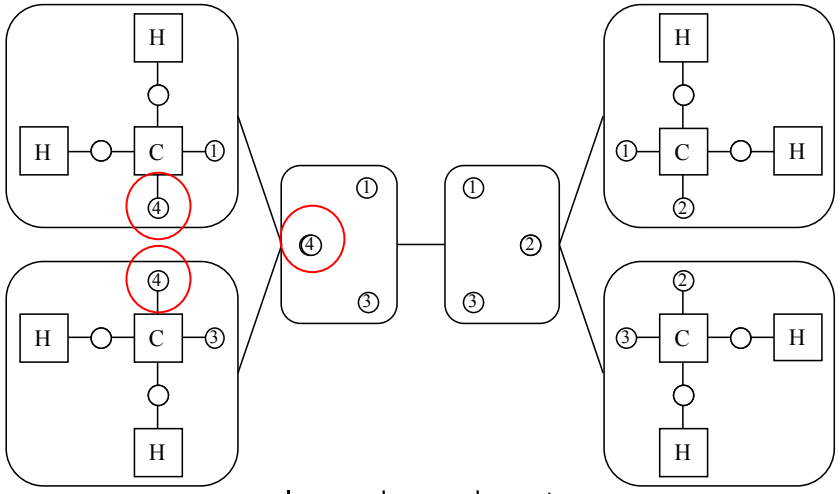
- HRG inference algorithm [Aguiñaga+, 16]
  - **Input:** Set of hypergraphs
  - **Output:** HRG w/ the following properties:
    - All of the input hypergraphs are in the language 😀
    - Guarantee the valence conditions 😀
    - No guarantee on 2-regularity 😭



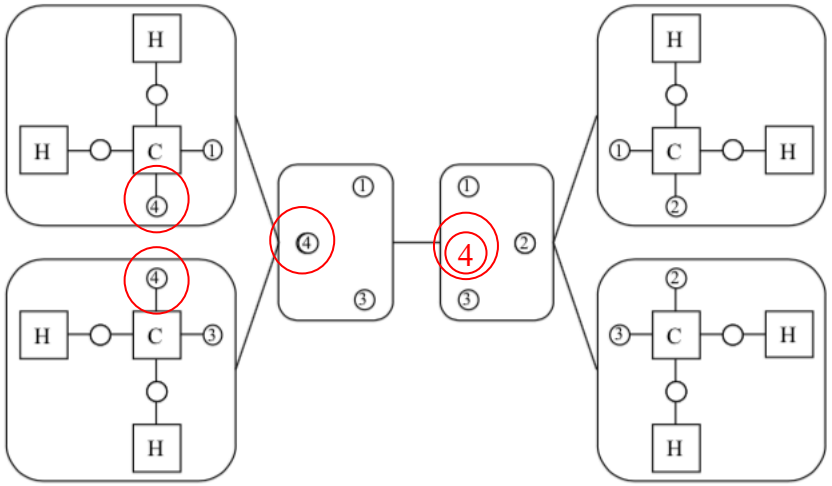
This cannot be transformed into a molecular graph

Irredundant tree decomposition is a key to guarantee 2-regularity

- Irredundant tree decomposition
  - The connected subgraph induced by a node must be a path
  - Any tree decomposition can be made irredundant in poly-time

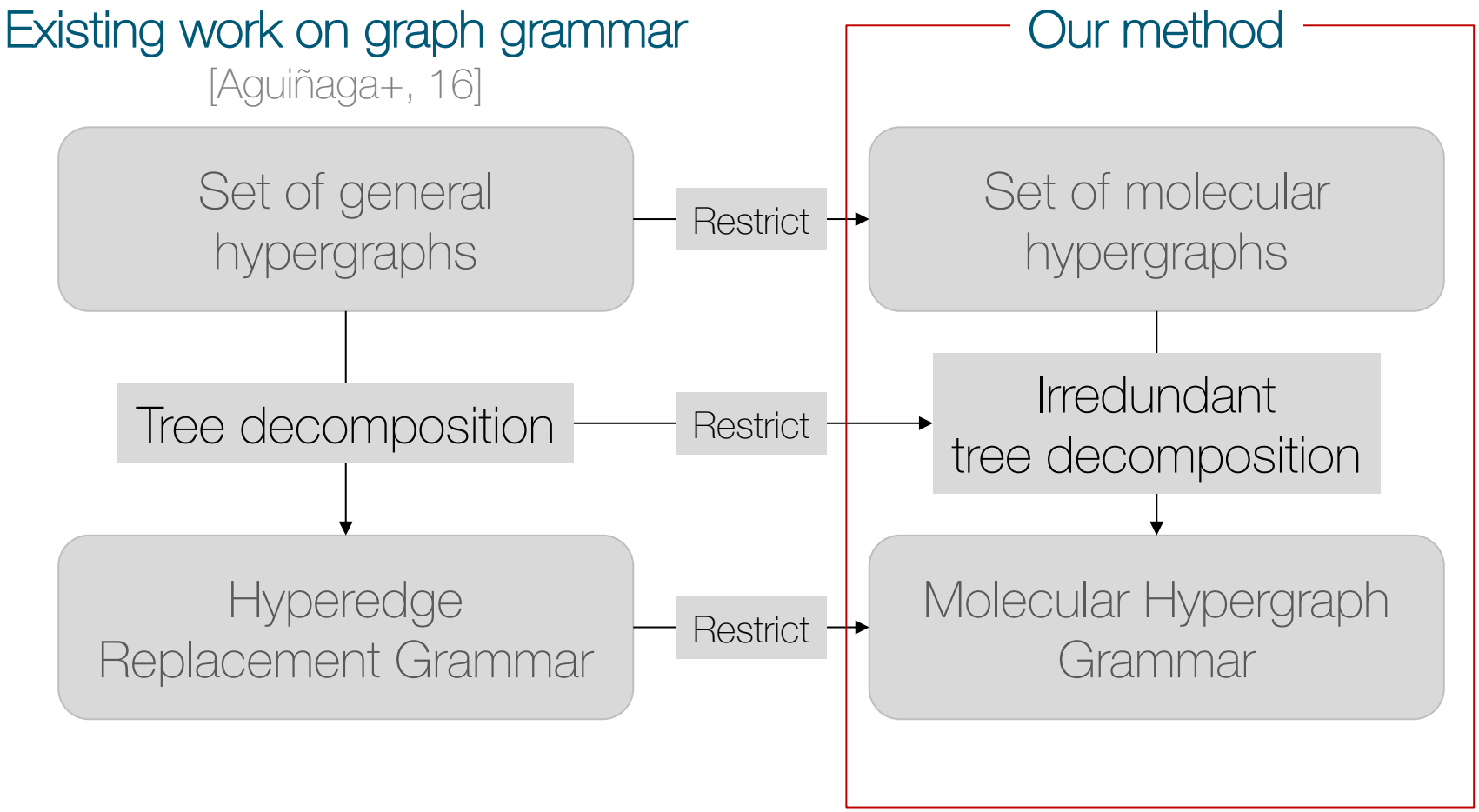


Irredundant



Redundant

Each restriction is our contribution in the literature of graph grammar



* HRG is a context-free grammar generating hypergraphs

Molecular hypergraph is used to satisfy the valence conditions, and irreduntant tree decomposition guarantees 2-regularity.

- MHG Inference algorithm
  - **Input:** Set of molecular graphs
  - **Output:** MHG w/ the following properties:
    - All of the input hypergraphs are in the language 😀
    - Guarantee the valence conditions 😀
    - Guarantee 2-regularity 🥳

Thanks to HRG

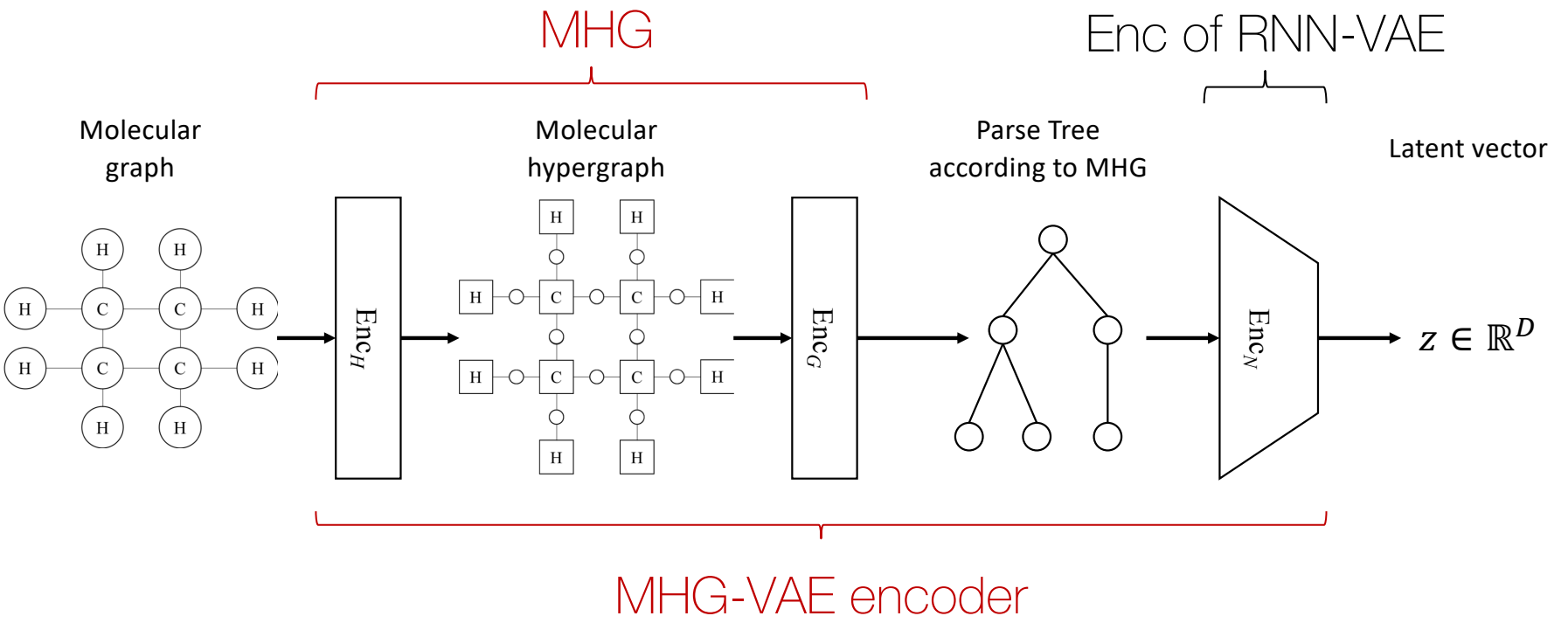
Our contribution

1. Convert molecular graphs into molecular hypergraphs
2. Compute tree decompositions of molecular hypergraphs
3. Convert each tree decomposition to be irredundant
4. Extract production rules
5. Compose MHG by taking their union

Application to Molecular Optimization

We obtain (Enc, Dec) between molecule and latent vector by combining MHG and RNN-VAE

- MHG-VAE: (Enc, Dec) between molecule & latent vector



First, we learn (Enc, Dec) between a molecule and its vector representation using MHG-VAE

- Global molecular optimization
  - **Find:** Molecule that maximizes the target
  - **Method:** VAE+BO
    1. Obtain MHG from the input molecules
    2. Train RNN-VAE on syntax trees
    3. Obtain vector representations $\{\boldsymbol{z_n} \in \mathbb{R}^D\}_{n=1}^N$

       Some of which have target values $\{y_n \in \mathbb{R}\}$
    4. BO gives us candidates $\{\boldsymbol{z}_m \in \mathbb{R}^D\}_{m=1}^M$ that may maxim
    5. Decode them to obtain molecules $\{G_m\}_{m=1}^M$

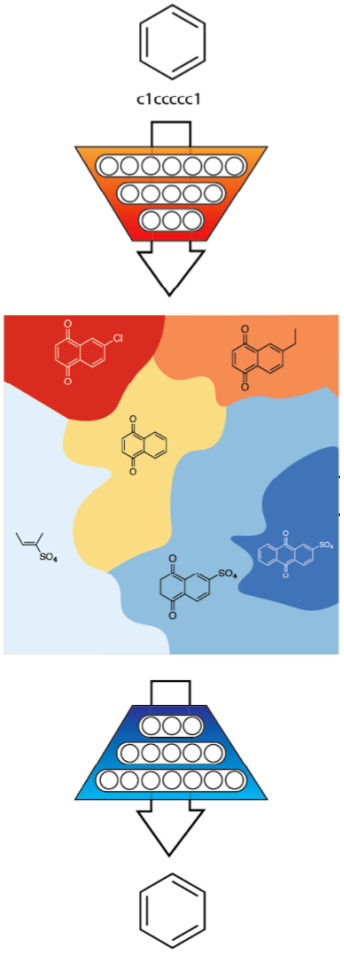


Image from [Gómez-Bombarelli+, 16]

Given vector representations and their target values, we use BO to obtain a vector that optimizes the target

- Global molecular optimization
  - –**Find:** Molecule that maximizes the target
  - –**Method:** VAE+BO
    1. Obtain MHG from the input molecules
    2. Train RNN-VAE on syntax trees
    3. Obtain vector representations $\{\boldsymbol{z}_n \in \mathbb{R}^D\}_{n=}^N$

       Some of which have target values $\{y_n \in \mathbb{R}\}$
    4. BO gives us candidates $\{\boldsymbol{z}_m \in \mathbb{R}^D\}_{m=1}^M$ that may maximize the target
    5. Decode them to obtain molecules $\{G_m\}_{m=1}^M$

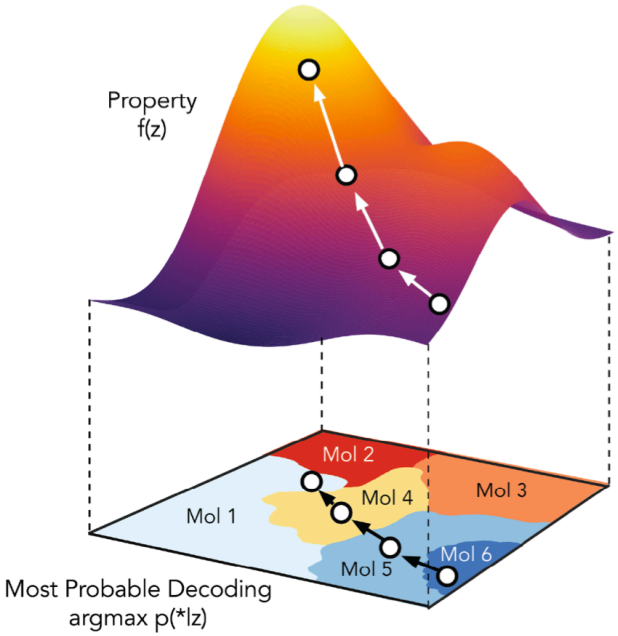


Image from [Gómez-Bombarelli+, 16]

We wish to understand the dependency of the performance on # of evaluations of function, $f: \mathbf{Molecule} \longmapsto \mathbf{Property}$

- Purpose of our empirical studies

  Validate the following questions:

  1. MHG improves the performance compared to JT-VAE?
  2. If # of evaluations is not limited, RL will be better than VAE
  3. If # of evaluations is limited, VAE will be better than RL

$$f(m) = \widehat{\log \mathrm{P}}(m) - \widehat{\mathrm{SA}}(m) - \widehat{\mathrm{cycle}}(m)$$

Water solubility

Synthetic accessibility score

Penalty to a ring larger than six

MHG achieves higher performance than the other VAE-based methods 😄

- Purpose of our empirical studies

  Validate the following questions:

  1. MHG improves the performance compared to VAE-based ones?
  2. If # of evaluations is not limited, RL will be better than VAE
  3. If # of evaluations is limited, VAE will be better than RL

Method	% Reconst.	Valid prior	Unlimited oracle case				
			1st	2nd	3rd	50th	Top 50 Avg.
CVAE	44.6%	0.7%	1.98	1.42	1.19	–	–
GVAE	53.7%	7.2%	2.94	2.89	2.80	–	–
SD-VAE	76.2%	43.5%	4.04	3.50	2.96	–	–
JT-VAE	76.7%	100%	5.30	4.93	4.49	3.48	3.93
GCPN	–	–	7.98	7.85	7.80	–	–
Ours	94.8%	100%	5.56	5.40	5.34	4.12	4.49

JT-VAE by Jin et al., ICML '18 (VAE+BO)
GCPN by You et al., NeurIPS '18 (RL)

But RL-based method is better than ours when # of evaluations is not limited🙃

- Purpose of our empirical studies

  Validate the following questions:

  1. MHG improves the performance compared to VAE-based ones?
  2. If # of evaluations is not limited, RL will be better than VAE
  3. If # of evaluations is limited, VAE will be better than RL

Method	% Reconst.	Valid prior	Unlimited oracle case				
			1st	2nd	3rd	50th	Top 50 Avg.
CVAE	44.6%	0.7%	1.98	1.42	1.19	–	–
GVAE	53.7%	7.2%	2.94	2.89	2.80	–	–
SD-VAE	76.2%	43.5%	4.04	3.50	2.96	–	–
JT-VAE	76.7%	100%	5.30	4.93	4.49	3.48	3.93
GCPN	–	–	7.98	7.85	7.80	–	–
Ours	94.8%	100%	5.56	5.40	5.34	4.12	4.49

JT-VAE by Jin et al., ICML '18 (VAE+BO)
GCPN by You et al., NeurIPS '18 (RL)

Ours is better than the RL-based method when # of evaluations is limited to 500 😄

- Purpose of our empirical studies

  Validate the following questions:

  1. MHG improves the performance compared to VAE-based ones?
  2. If # of evaluations is not limited, RL will be better than VAE
  3. If # of evaluations is limited, VAE will be better than RL

Method	Limited oracle case				
	1st	2nd	3rd	50th	Top 50 Avg.
JT-VAE	1.69	1.68	1.60	-9.93	-1.33
GCPN	2.77	2.73	2.34	0.91	1.36
Ours	5.24	5.06	4.91	4.25	4.53

JT-VAE by Jin et al., ICML '18 (VAE+BO)
GCPN by You et al., NeurIPS '18 (RL)

We develop a graph-grammar based molecular representation, MHG, as well as its combination with VAE.

- We develop a molecular hypergraph grammar (MHG)
  - Molecules generated by MHG always satisfy the valence conditions
    - Hypergraph representation
    - Irredundant tree decomposition
  - Inference algorithm from data
    - No need to write rules by hand
    - The inferred MHG can describe all of the input data
- We apply MHG to molecular optimization
  - MHG-VAE > JT-VAE
  - MHG-VAE > GCPN when # of evals is limited

[Gómez-Bombarelli+, 16] Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. Automatic chemical design using a data-driven continuous representation of molecules. ACS Central Science, 2018. (ArXiv ver. appears in 2016)

[Jin+, 18] Jin, W., Barzilay, R., and Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In Proceedings of the Thirty-fifth International Conference on Machine Learning, 2018.

[You+, 18] You, J., Liu, B., Ying, Z., Pande, V., and Leskovec, J. Graph convolutional policy network for goal-directed molecular graph generation. In Advances in Neural Information Processing Systems 31, pp. 6412–6422, 2018.

[Aguiñaga+, 16] Aguiñaga, S., Palacios, R., Chiang, D., and Weninger, T. Growing graphs from hyperedge replacement graph grammars. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 469–478, 2016.