Feature Extraction from Bounded Tree-Width Graphs Using Canonical String Representations

Master's Thesis

Ivan Danielov Ivanov

University of Bonn

Supervisor:
Dr. Tamás Horváth

Reviewers:
Prof. Dr. Jens Lehmann
Prof. Dr. rer. nat. Stefan Wrobel

October 5, 2016

Outline

Outline

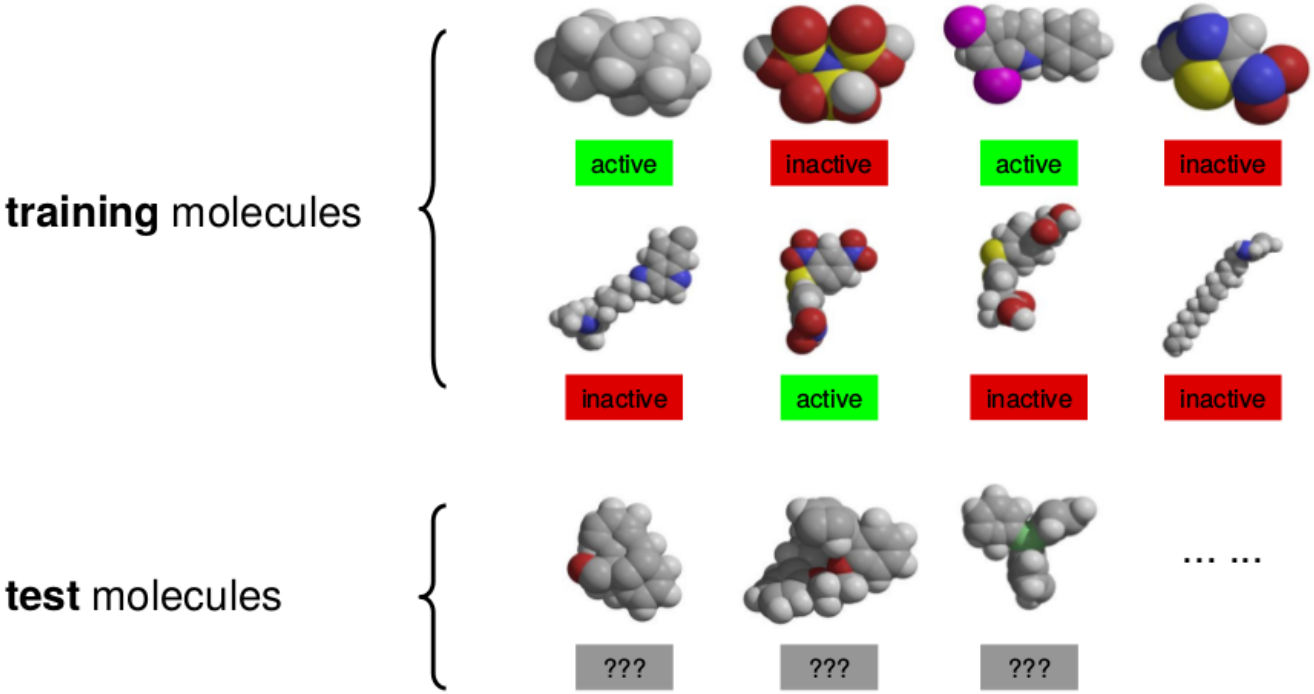
Mining and learning in graphs

Some applications:

- Prediction of properties of chemical compounds
- Type inference on RDF data
- Predicting behavior on social networks
- Classifying pages on the web graph
- On-line fraud detection
- ...

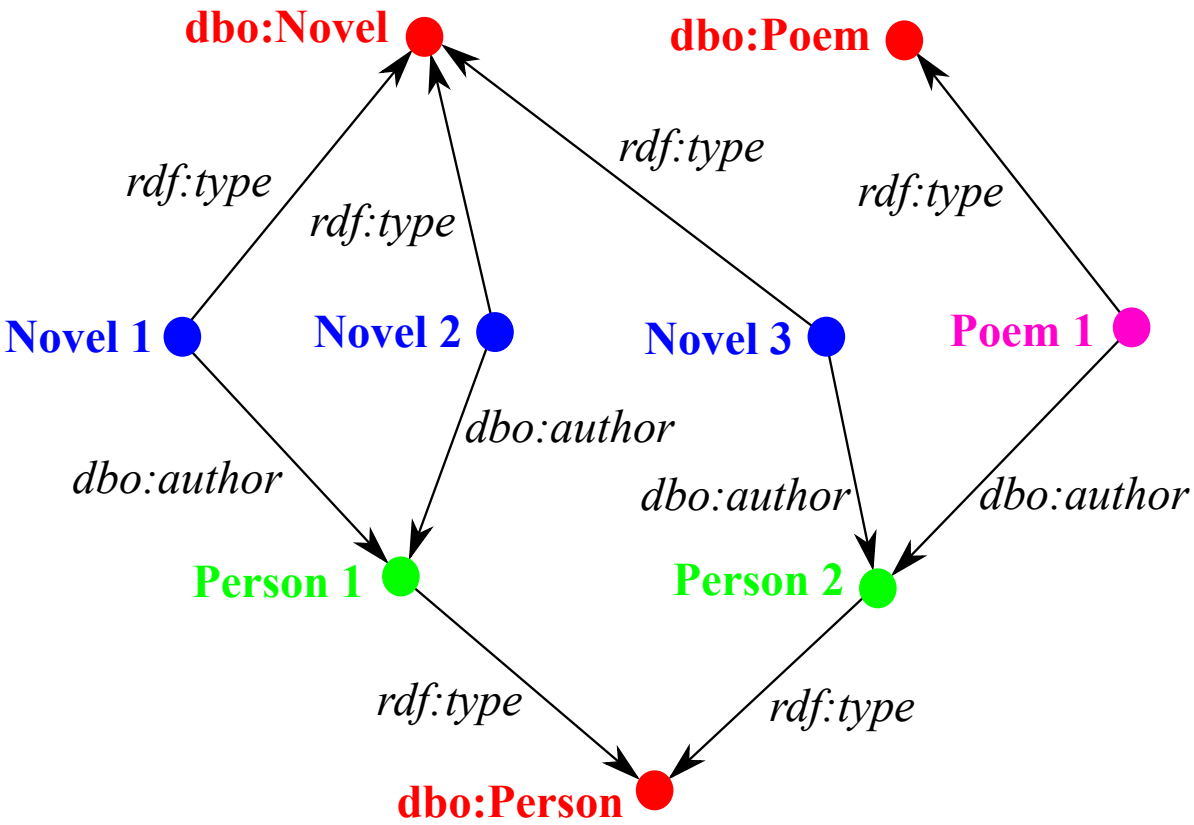
Example: Virtual screening in drug discovery

- select a limited number of candidate compounds from **millions** of database molecules that are most likely to possess a desired biological activity



Copyright: Prof. Stefan Wrobel

Example: Similarity of RDF resources



How similar are Person 1 and Person 2?

Importance of learning in graphs

- Reducing costs for discovering properties of chemical compounds
- More stable Semantic Web data inference compared to logic-based reasoning

Challenges of learning in graphs

- No single fixed-width table or single row representation

  $\Longrightarrow$ Cannot apply learning algorithms directly

- Cannot decide if two graphs are the same
  - Graph isomorphism is not known to be in P

Main contribution

- **Feature extraction** from graphs
  - Useful e.g. for graph kernels

Main contribution

- **Feature extraction** from graphs
  - Useful e.g. for graph kernels
- Similarity of **canonical (string) representations**
  - w-**shingles** of the canonical string representations

Main contribution

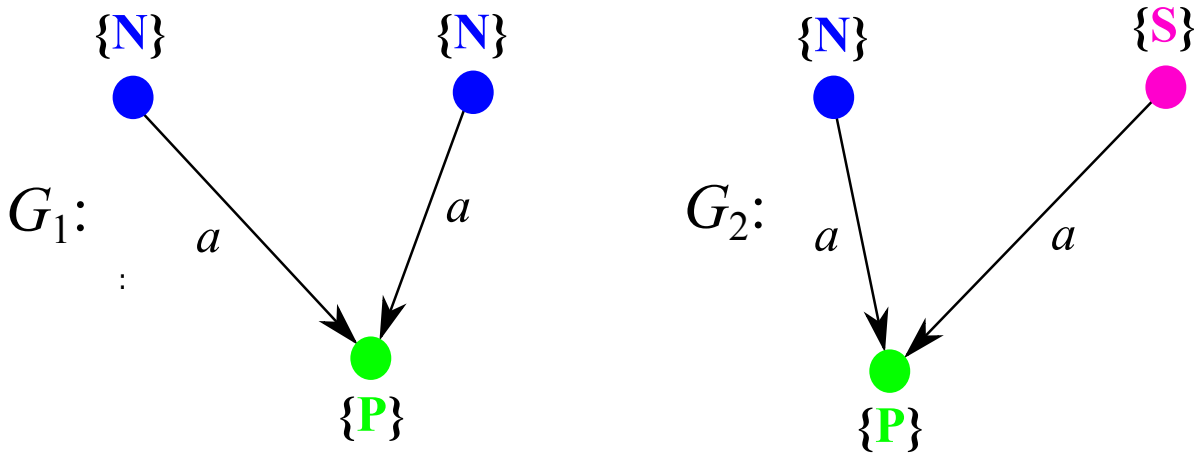
- **Feature extraction** from graphs
  - Useful e.g. for graph kernels
- Similarity of **canonical (string) representations**
  - w-**shingles** of the canonical string representations
- Can't compute canonical representations for **general** graphs

Main contribution

- **Feature extraction** from graphs
  - Useful e.g. for graph kernels
- Similarity of **canonical (string) representations**
  - w-**shingles** of the canonical string representations
- Can't compute canonical representations for **general** graphs
- Graphs of bounded **tree-width**

Main contribution

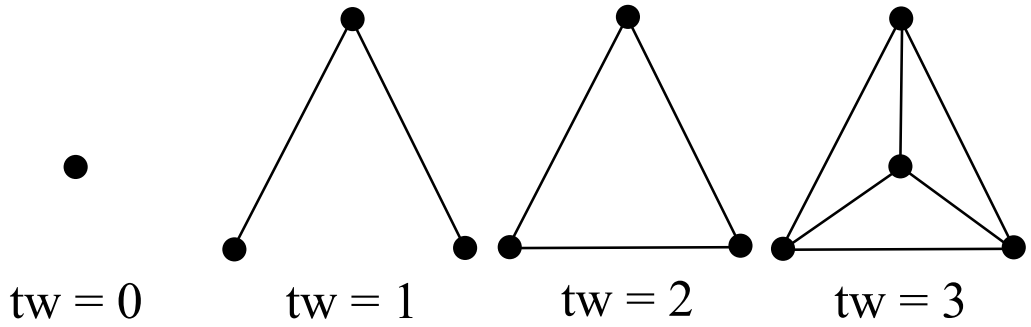
- **Feature extraction** from graphs
  - Useful e.g. for graph kernels
- Similarity of **canonical (string) representations**
  - w-**shingles** of the canonical string representations
- Can't compute canonical representations for **general** graphs
- Graphs of bounded **tree-width**



$C(G_1) =$ "(0.1;(1.2;(a,((1,0))),N),(1.2;(a,((1,0))),N),P)"
$C(G_2) =$ "(0.1;(1.2;(a,((1,0))),N),(1.2;(a,((1,0))),S),P)"

Tree-width

- **Tree-width** measures the **tree-likeness** of a graph
- Graph class of tree-width up to 3:
  - **Canonical string representations** computed in linear time
  - **Isomorphism** and **membership** decidable in linear time
  - Practical applications:
    - Most **molecular graphs** have tree-width at most 2 (Horváth, Ramon, and Wrobel, 2010)
    - The **direct neighborhoods** of RDF nodes have tree-width 1

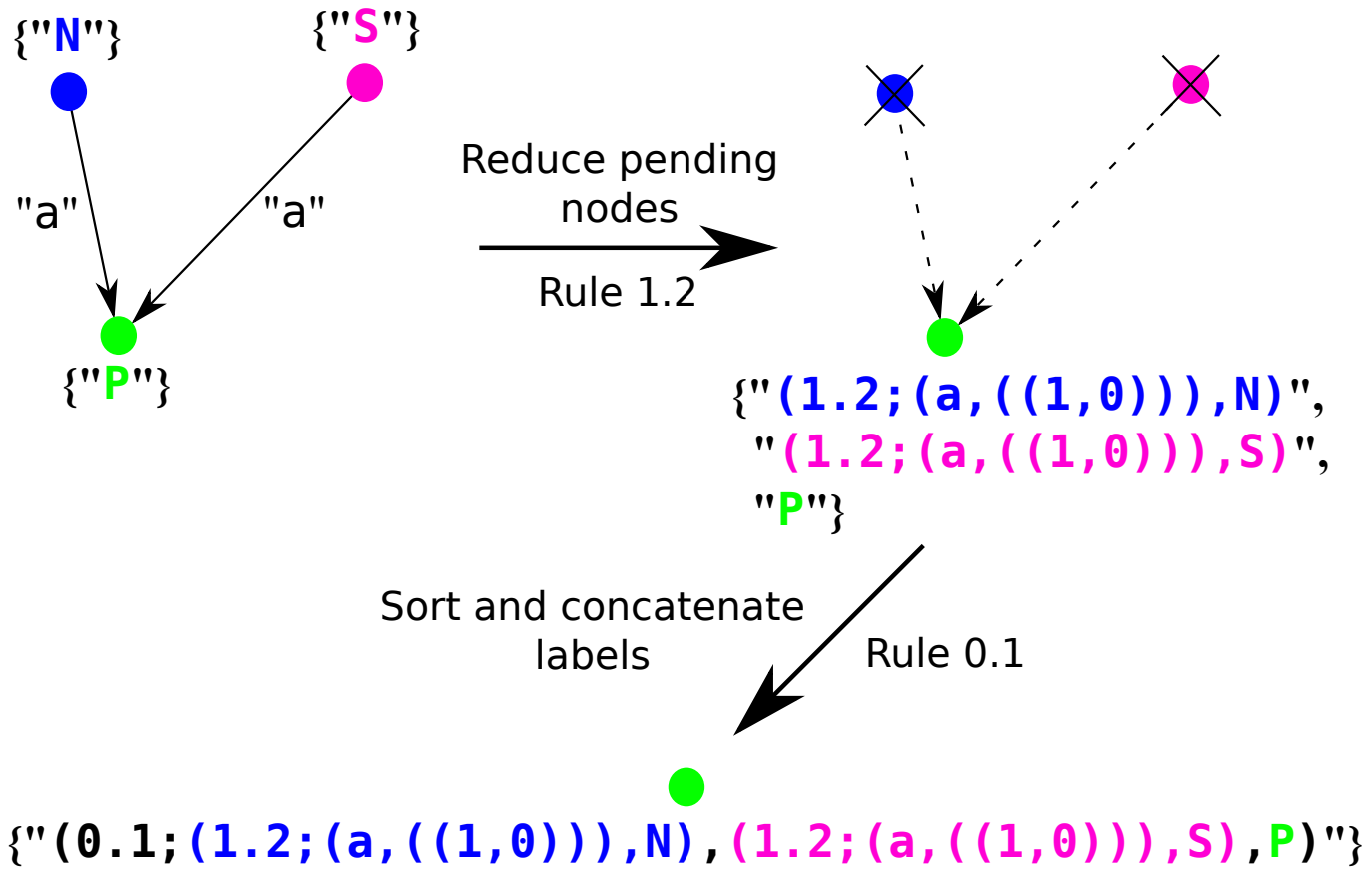


Outline

Canonical representations of bounded tree-width graphs

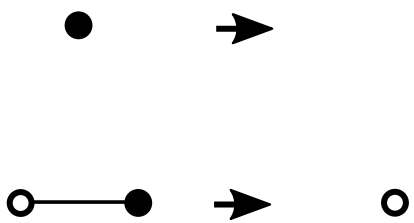
- Arnborg and Proskurowski (1992)
- Reduce the graph to single vertex using **safe reduction rules**
  - A graph has tree-width ≤ 3 if it is reduced to single vertex
- In total 24 reduction rules (isolates $+$ conflicts) for tree-width ≤ 3

Example for computing a canonical string representation

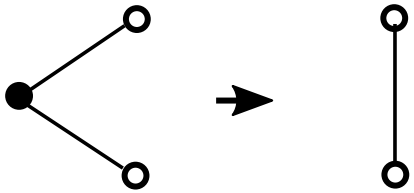


The label of the last vertex is the **canonical string representation**

Safe reduction rules



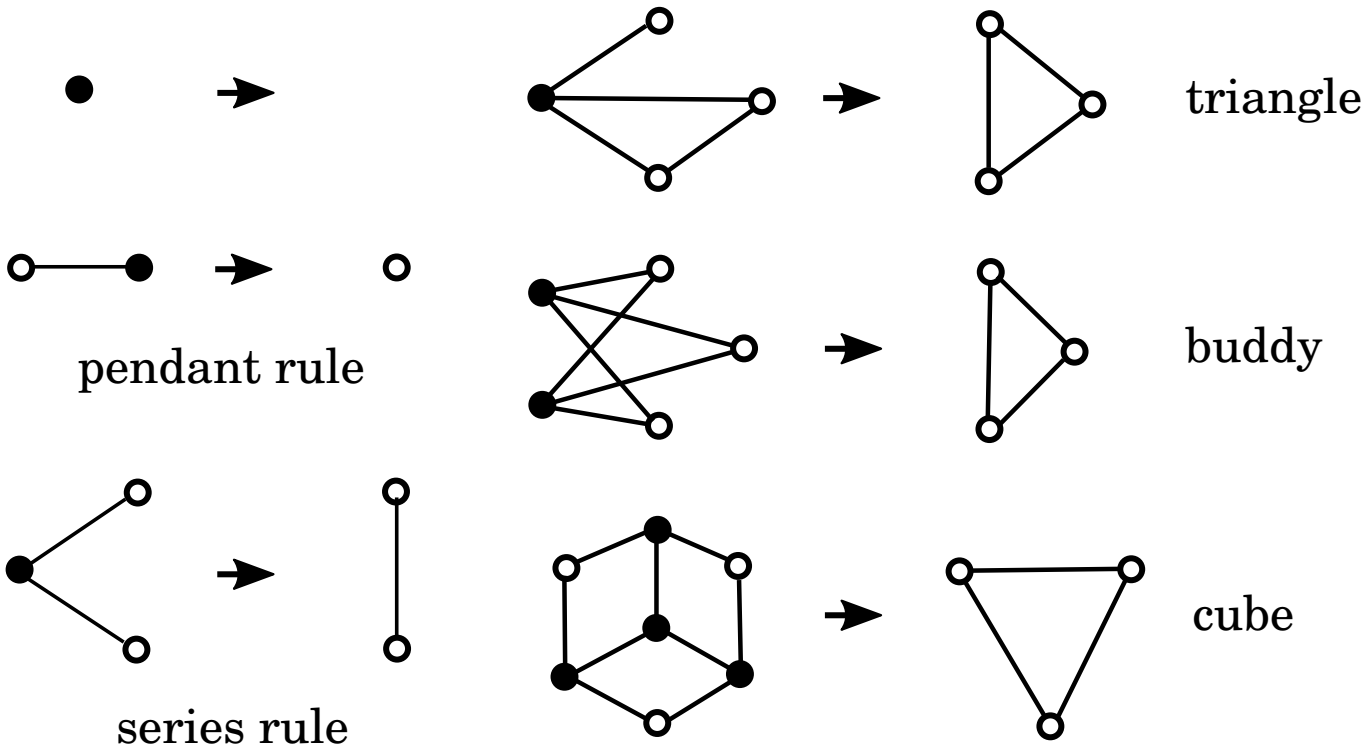
pendant rule



series rule

Copyright: Arnborg and Proskurowski (1992)

Safe reduction rules



Copyright: Arnborg and Proskurowski (1992)

Similarity of canonical strings

- Map each string to a set of w-shingles (Broder, 2000)
- Define the features as the Rabin's fingerprints of the w-shingles
- Compute the similarity between the feature sets obtained
- Example:

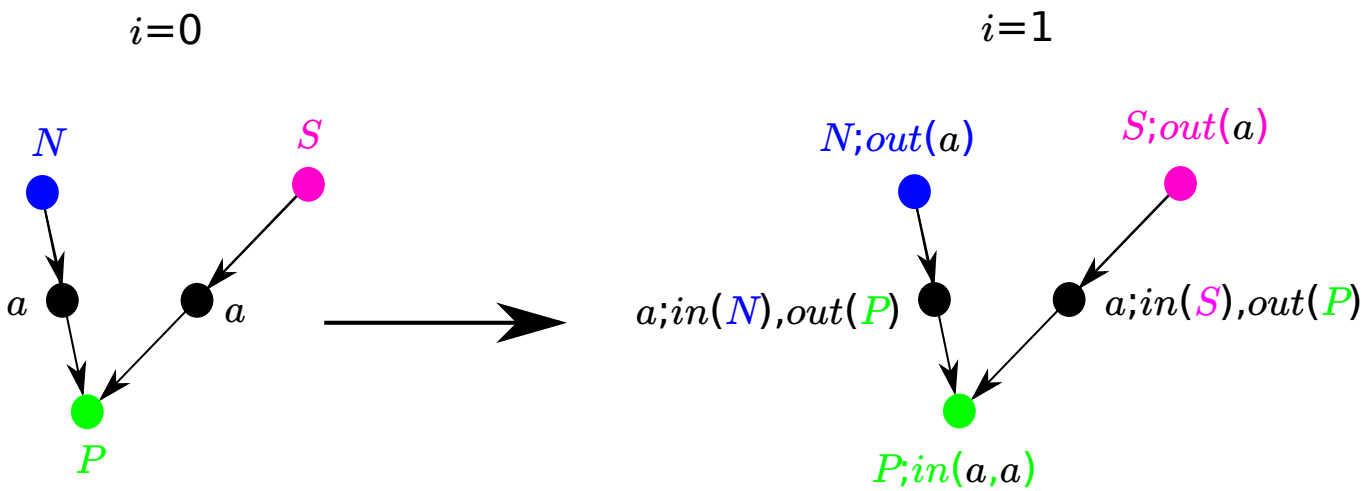
$$S_1 = \texttt{w\_shingles}(\text{“centre”}, 3) = \{\text{“cen”}, \text{“ent”}, \text{“ntr”}, \text{“tre”}\}$$

$$S_2 = \texttt{w\_shingles}(\text{“center”}, 3) = \{\text{“cen”}, \text{“ent”}, \text{“nte”}, \text{“ter”}\}$$

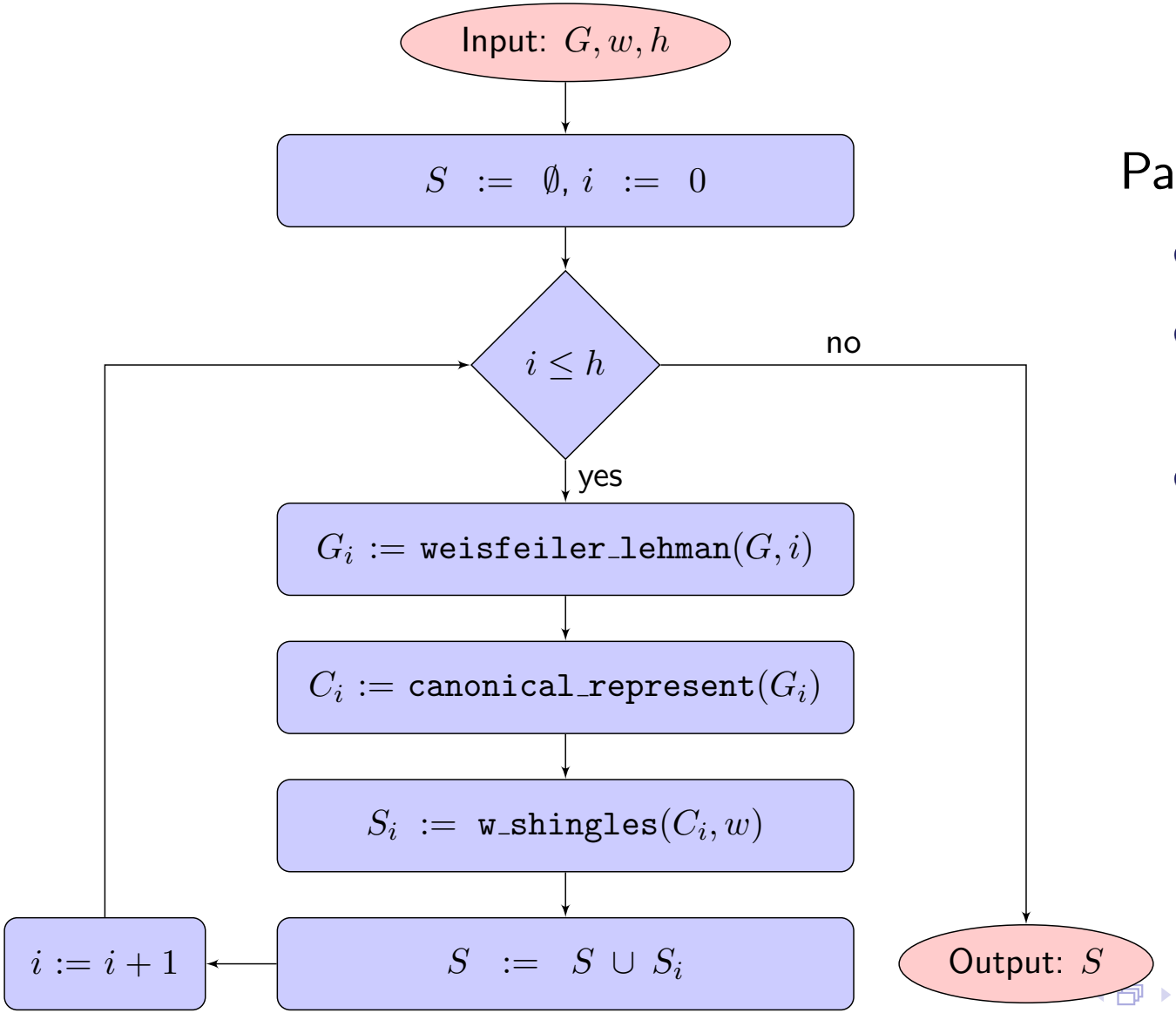
$$\texttt{Sim}_{\text{Jaccard}}(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = \frac{1}{3}$$

Refining the labels in the graph

- Refine the labels using the **Weisfeiler-Lehman** test of isomorphism (Weisfeiler and Lehman, 1968)
  $\Longrightarrow$ Leads to larger distances between dissimilar graphs
- Example:



Algorithm for computing the features



Parameters:

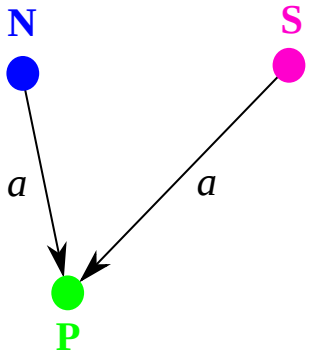
- Graph G
- Shingling window $w > 0$
- WL iterations $h \geq 0$

Feature extraction example

Let $w = 5$ and $h = 1$

$i = 0$

$G_0 = G$:

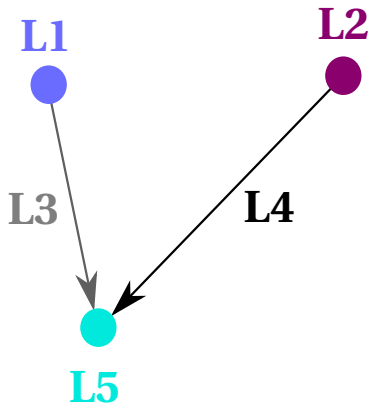


$C_0 =$ "(0.1;(1.2;(a,((1,0)))
,N),(1.2;(a,((1,0))),S),P)"

$S_0 = \{$"(0.1;", "0.1;(", ...$\}$

$i = 1$

G_1 :



$C_1 =$ "(0.1;(1.2;(L3,((1,0))),
L1),(1.2;(L4,((1,0))),L2),L5)"

$S_1 = \{$"(0.1;", "0.1;(", ...$\}$

$$S = S_0 \cup S_1$$

Properties of the method

- Some nice properties:
  - Performs in **linear** time
  - Operates on an expressive graph class
  - Provides the computed features explicitly
- However, non-injective modulo isomorphism
  - Not really a practical disadvantage

Outline

Problems for evaluation

- Classification of chemical compounds
  - Ordinary graphs
  - RDF
- Predicting the types of RDF resources in DBpedia

Chemical datasets

- NCI-HIV (about 40 000 compounds, skewed classes)
  - Problem: CA vs CM
  - Problem: CA + CM vs CI
  - Problem: CA vs CI
- Mutagenesis (188 compounds)
- Carcinogenesis (340 compounds)
  - Problem: CAR-337 (298 + 39 from PTE-1 challenge)
  - Problem: MUTAG (340 compounds)

Evaluation on chemical datasets (ordinary graphs)

- Classifier: SVM with RBF kernel on the binary feature vectors
- Scoring: 10-fold cross-validation

Problem (score: AUC)	Our method	NSPDK[1]
NCI-HIV (CA vs CM)	**.842(.032)**	.841(.480)
NCI-HIV (CA + CM vs CI)	.841(.010)	**.849(.210)**
NCI-HIV (CA vs CI)	.949(.028)	**.956(.130)**

Problem (score: accuracy)	Our method	Lodhi and Muggleton (2005)
Mutagenesis (188)	.942(.036)	**.958(.330)**

Similar scores, our method performs in **linear** time (theoretically guaranteed) and shows **less variance**.

[1] NSPDK (Costa and Grave, 2010)

Preprocessing of the RDF data

- General RDF preprocessing:
  - Each **vertex** has as set of labels its **RDF types**
  - Each **edge** has as label its **predicate**
- For chemical RDF datasets:
  - **Pre-1**: Only the molecule structure
  - **Pre-2**: The molecule structure and the boolean literals

Evaluation on chemical datasets (RDF)

- Classifier: SVM with RBF kernel on the binary feature vectors
- Scoring: accuracy, 10-fold cross-validation

Problem	Our method (Pre-1)	Our method (ordinary graphs)
Mutagenesis (188)	.921(.057)	**.942(.036)**

Problem	Our method (Pre-1)	Our method (Pre-2)	DL-learner[2]	BoL kernel[3]
Carcinogenesis (CAR-337)	.644(.111)	**.815(.157)**	.674(.790)	-
Carcinogenesis (MUTAG)	.833(.076)	.773(.086)	-	**.951(.004)**

Similar scores between RDF and ordinary graphs.
Significant results for **CAR-337** using **Pre-2** preprocessing.

[2] DL-learner (Lehmann, 2009)
[3] BoL kernel (de Vries and de Rooij, 2015)

Evaluation on DBpedia

- Problem: Predict the types of RDF resources in DBpedia (multi-label classification)
- Sample 10 000 resources with incoming degree ≥ 25
- Preprocess RDF data and extract direct neighborhoods of the resources
- Train SVM with RBF or linear kernel and evaluate F-measure over 10-fold cross-validation

Neighborhood type	Our method (RBF kernel)	Our method (linear kernel)	SDType[4]
in	.895(.005)	.887(.006)	**.899**

Our **generally** applicable method performs similarly to the **RDF specific** method SDType.

[4] SDType (Paulheim and Bizer, 2013)

Outline

Summary

- Fast and simple feature extraction method
  - Canonical string representations
  - Restricted to graphs of tree-width up to 3
  - Large overlap of the strings means similar graphs
  - Conceptually different from most other graph kernels
- Empirical evaluation results
  - Very good performance on classifying chemical compounds
  - Similar to RDF specific approaches performance on predicting types of RDF resources
  - Preprocessing may have large impact on the results

Ideas for future work

- Feature selection before applying machine learning
- Trying different classification models
- New preprocessing techniques for RDF
- Conducting experiments on more learning problems
- Computing approximate canonical strings of graphs with larger tree-width