Ring Based Approximation of Graph Edit Distance

Presented at S+SSPR 2018, Beijing, China, August 17–19, 2018

D. B. Blumenthal[1], S. Bougleux[2], J. Gamper[1], L. Brun[2]

[1]Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy

[2]Normandie Université, UNICAEN, ENSICAEN, CNRS, GREYC, Caen, France

August 18, 2018

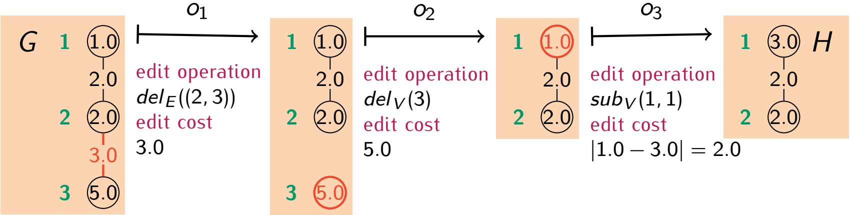
Background

Graph Edit Distance (Definition)

- idea: distance between labeled graphs G and H = minimal amount of distortion needed for transforming G into H
- edit operations and edit costs:
  - substituting a node $u \in V^G$ by a node $v \in V^H \rightsquigarrow c_V(u, v)$
  - deleting an isolated node $u \in V^G \rightsquigarrow c_V(u, \epsilon)$
  - inserting an isolated node $v \in V^H \rightsquigarrow c_V(\epsilon, v)$
  - substituting an edge $e \in E^G$ by an edge $f \in E^H \rightsquigarrow c_E(e, f)$
  - deleting an edge $e \in E^G \rightsquigarrow c_E(e, \epsilon)$
  - inserting an edge $f \in E^H \rightsquigarrow c_E(\epsilon, f)$
- sequence $P = (o_i)_{i=1}^r$ of edit operations is edit path between G and H iff $(o_r \circ \ldots \circ o_1)(G) = H \rightsquigarrow c(P) = \sum_{i=1}^r c(o_i)$
- $\text{GED}(G, H) := \min\{c(P) \mid P \text{ is edit path between } G \text{ and } H\}$
- computing GED is NP-hard $\rightsquigarrow$ approximative techniques needed

Graph Edit Distance (Example)

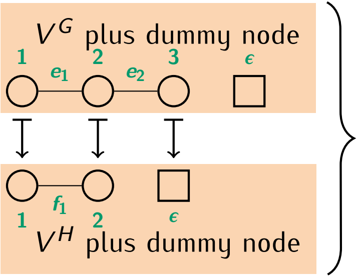
- real-valued, positive node and edge labels:
  - $\ell_V^G : V^G \to \mathbb{R}_{\geq 0}$, $\ell_V^H : V^H \to \mathbb{R}_{\geq 0}$
  - $\ell_E^G : E^G \to \mathbb{R}_{\geq 0}$, $\ell_E^H : E^H \to \mathbb{R}_{\geq 0}$
- edit costs:
  - $c_V(u, v) = |\ell_V^G(u) - \ell_V^H(v)|$, $c_V(u, \epsilon) = \ell_V^G(u)$, $c_V(\epsilon, v) = \ell_V^H(v)$
  - $c_E(e, f) = |\ell_E^G(e) - \ell_E^H(f)|$, $c_E(e, \epsilon) = \ell_E^G(e)$, $c_E(\epsilon, f) = \ell_E^H(f)$
  - $c(P) = 3.0 + 5.0 + 2.0 = 10$



Graph Edit Distance (Computation)

- if we know how to edit nodes $u_1, u_2 \in V^G$, then we know how to edit the edge $e = (u_1, u_2) \in E^G$

$\Rightarrow$ complete set of node operations induces edit path

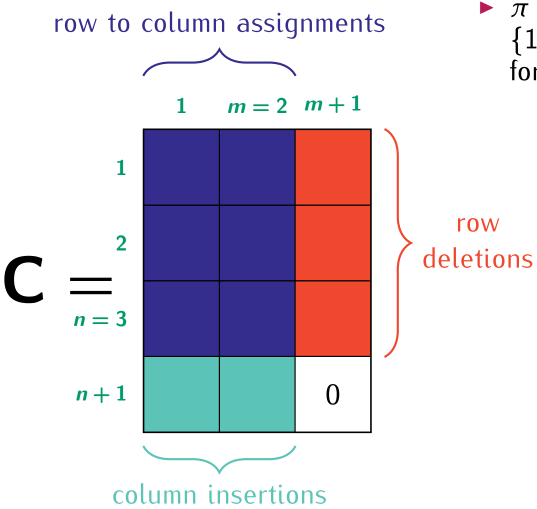


induced edge edit operations:

- edge e_1 is substituted by edge f_1
- edge e_2 is deleted

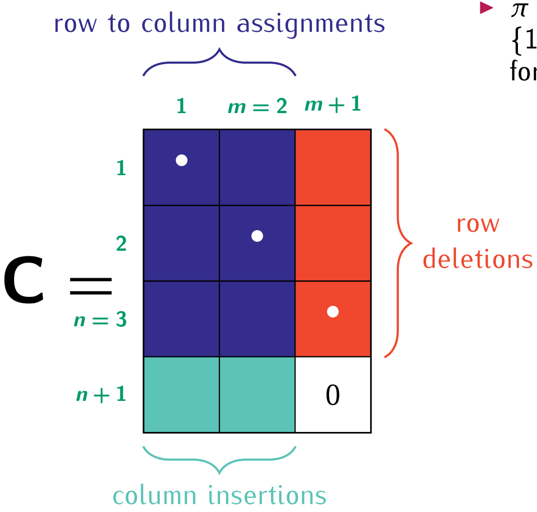
- task: find complete set of node edit operations that induces cheap edit path

Linear Sum Assignment with Error Correction



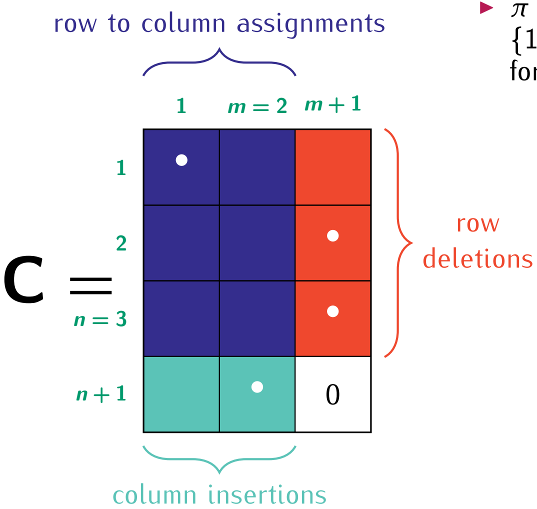
- $\pi \in \{1, \ldots, n+1\} \times \{1, \ldots, m+1\}$ is solution for LSAPE instance $\mathbf{C}$ iff:
  - each row except for $n+1$ is covered exactly once
  - each column except for $m+1$ is covered exactly once
  - solution π minimizing $\mathbf{C}(\pi) = \sum_{i=1}^{n+1} \sum_{k \in \pi[i]} c_{i,k}$ can be computed in $O(\min\{n, m\}^2 \max\{n, m\})$ time, greedy suboptimal solutions in $O(nm)$ time

Linear Sum Assignment with Error Correction



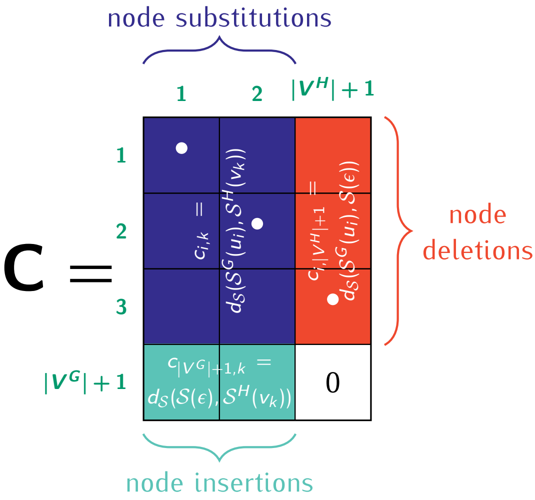
- $\pi \in \{1, \ldots, n+1\} \times \{1, \ldots, m+1\}$ is solution for LSAPE instance $\mathbf{C}$ iff:
  - each row except for $n+1$ is covered exactly once
  - each column except for $m+1$ is covered exactly once
  - solution π minimizing $\mathbf{C}(\pi) = \sum_{i=1}^{n+1} \sum_{k \in \pi[i]} c_{i,k}$ can be computed in $O(\min\{n, m\}^2 \max\{n, m\})$ time, greedy suboptimal solutions in $O(nm)$ time

Linear Sum Assignment with Error Correction



- $\pi \in \{1, \ldots, n+1\} \times \{1, \ldots, m+1\}$ is solution for LSAPE instance $\mathbf{C}$ iff:
  - each row except for $n+1$ is covered exactly once
  - each column except for $m+1$ is covered exactly once
  - solution π minimizing $\mathbf{C}(\pi) = \sum_{i=1}^{n+1} \sum_{k \in \pi[i]} c_{i,k}$ can be computed in $O(\min\{n, m\}^2 \max\{n, m\})$ time, greedy suboptimal solutions in $O(nm)$ time

LSAPE Based Heuristics for GED (Paradigm)



- solution for LSAPE instance **C** $\widehat{=}$ complete set of node operations $\widehat{=}$ edit path between G and H $\rightsquigarrow$ upper bound for GED
- $\mathcal{S}^{G|H}(\cdot)$: local structure rooted at node of one of the input graphs
- $d_{\mathcal{S}}$: distance measure for local structures

LSAPE Based Heuristics for GED (Instantiations)

$\mathcal{S}^G(u) \mathrel{\widehat{=}}$ node u and its incident edges [2, 5]:

- baseline instantiation, yields rather loose upper bound
- construction time for **C** cubic or quadratic in maximum degrees (depending on distance measure for local structures)

$\mathcal{S}^G(u) \mathrel{\widehat{=}}$ subgraph of radius L rooted at u [3]:

- yields tighter upper bound than baseline
- construction time for **C** polynomially bounded only for graphs with constantly bounded maximum degrees

$\mathcal{S}^G(u) \mathrel{\widehat{=}}$ walks of length L rooted at u [4]:

- yields tighter upper bound than baseline
- construction time for **C** bounded by polynomial of degree $2 + \omega$ (ω is matrix multiplication complexity exponent)
- suffers from tottering and supports only constant edit costs

Rings as Local Structures

Towards a New LSAPE Based Heuristic for GED

Shortcomings of Local Structures Used in Existing Instantiations

- baseline (root plus incident edges): considers only very local information $\rightsquigarrow$ loose upper bound on some datasets
- subgraph of fixed radius around root: construction of **C** prohibitively expensive
- walks of fixed length starting at root: tottering, supports only constant edit costs $\rightsquigarrow$ loose upper bound on some datasets

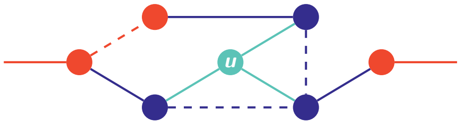
Desiderata

- define ring structures $\mathcal{S}^G(u) = \mathcal{R}_L^G(u)$ and distance measure $d_\mathcal{S} = d_\mathcal{R}$, such that:
    - $\mathcal{R}_L^G(u)$ considers more information than the baseline and contains nodes and edges at most once to avoid tottering
    - $d_\mathcal{R}$ supports general edit costs and can be evaluated quickly to ensure reasonable construction time for **C**

Definition of Rings

- rings are sequences of layers: $\mathcal{R}_L^G(u) = (\mathcal{L}_l^G(u))_{l=0}^{L-1}$, $\mathcal{R}_L(\epsilon) = ((\emptyset, \emptyset, \emptyset)_l)_{l=0}^{L-1}$
- layers contain nodes, inner, and outer edges at distance l from root: $\mathcal{L}_l^G(u) = (V_l^G(u), OE_l^G(u), IE_l^G(u))$
- nodes at distance l from root are reachable from root on path of length l: $V_l^G(u) = \{u' \in V^G \mid d_V^G(u, u') = l\}$
- inner edges at distance l from root connect two nodes at distance l: $IE_l^G(u) = E^G \cap \left(V_l^G(u) \times V_l^G(u)\right)$
- outer edges at distance l from root connect nodes at distance l and $l+1$: $OE_l^G(u) = E^G \cap \left(V_l^G(u) \times V_{l+1}^G(u)\right)$

$\mathcal{R}_3^G(u)$



$\mathcal{L}_0^G(u)$
$\mathcal{L}_1^G(u)$
$\mathcal{L}_2^G(u)$

Definition of Distance Measure for Rings

- distance measure for rings:

$$d_{\mathcal{R}}\left(\mathcal{R}_L^G(u), \mathcal{R}_L^H(v)\right) = \sum_{l=0}^{L-1} \lambda_l d_{\mathcal{L}}\left(\mathcal{L}_l^G(u), \mathcal{L}_l^H(v)\right)$$

- $\lambda \in \mathbb{R}^L_{\geq 0}$: weights for distances between layers
- distance measure for layers:

$$d_{\mathcal{L}}\left(\mathcal{L}_l^G(u), \mathcal{L}_l^H(v)\right) = \alpha_0 \phi_V\left(V_l^G(u), V_l^H(v)\right) + \alpha_1 \phi_E\left(OE_l^G(u), OE_l^H(v)\right) + \alpha_2 \phi_E\left(IE_l^G(u), IE_l^H(v)\right)$$

- $\phi_V : \mathcal{P}(V^G) \times \mathcal{P}(V^H) \to \mathbb{R}_{\geq 0}$: distance measure for node sets
- $\phi_E : \mathcal{P}(E^G) \times \mathcal{P}(E^H) \to \mathbb{R}_{\geq 0}$: distance measure for edge sets
- $\alpha \in \mathbb{R}^3_{\geq 0}$: weights for distances between nodes, outer, and inner edges

Definitions of ϕ_V and ϕ_E

- node sets $U = \{u_1, \ldots, u_r\} \subseteq V^G$, $V = \{v_1, \ldots, u_s\} \subseteq V^H$
  $\rightsquigarrow$ define $\phi_V(U, V)$ as follows (definitions of ϕ_E analogous):
- LSAPE based approach:
  1. define instance $\mathbf{C} = (c_{i,k}) \in \mathbb{R}^{(r+1)\times(s+1)}$: $c_{i,k} = c_V(u_i, v_k)$, $c_{i,s+1} = c_V(i, \epsilon)$, and $c_{r+1,k} = c_V(\epsilon, v_k)$ for $i \neq r+1$, $k \neq s+1$
  2. compute solution π for $\mathbf{C}$, either optimally or greedily
  3. return $\phi_V(U, V) = \mathbf{C}(\pi)/\max\{r, s, 1\}$
- multiset intersection based approach:
  1. compute size Γ of intersection between multisets containing U's and V's node labels
  2. compute avg. insertion, deletion, and substitution costs c_V^{ins}, c_V^{del}, and c_V^{sub} between the nodes in U and V
  3. return $\phi_V(U, V) =$ $[c_V^{del}\delta_{r>s}(r-s)+c_V^{ins}\delta_{s>r}(s-r)+c_V^{sub}(\min\{r, s\}-\Gamma)]/\max\{r, s, 1\}$

Construction of Rings and Choice of Meta-Parameters

- construction of ring rooted at node $u \in V^G$: variant of BFS, runs in $O(|V^G| + |E^G|)$ time
- choice of weights α, λ, and number of layers L:
  1. sample training set $\mathcal{T}$ from graph database
  2. set L to upper bound for ring sizes, e. g., $L = 1 + \max_{G \in \mathcal{T}} |V^G|$
  3. compute all rings for all graphs contained in $\mathcal{T}$
  4. lower L to maximal size of the rings
  5. learn α and λ by calling blackbox optimizer with objective $obj(\alpha, \lambda) = [\mu + (1-\mu)\left(\frac{|\operatorname{supp}(\lambda)|-1}{\max\{1, L-1\}}\right)] \sum_{(G,H) \in \mathcal{T}^2} \mathtt{RING}_{\alpha,\lambda}^{\phi_V,\phi_E}(G, H)$ and the constraints that α and λ be simplex vectors
     - $\mathtt{RING}_{\alpha,\lambda}^{\phi_V,\phi_E}(G, H)$: upper bound for $\mathrm{GED}(G, H)$ given fixed choices of α, λ, ϕ_V, and ϕ_E
     - tuning parameter $\mu \in [0, 1]$: close to 1 $\rightsquigarrow$ optimize for accuracy, close to 0 $\rightsquigarrow$ optimize for runtime
  6. lower L to $L = 1 + \max_{l \in \operatorname{supp}(\lambda)} l$

Construction of Ring of Size L Rooted at $u \in V^G$

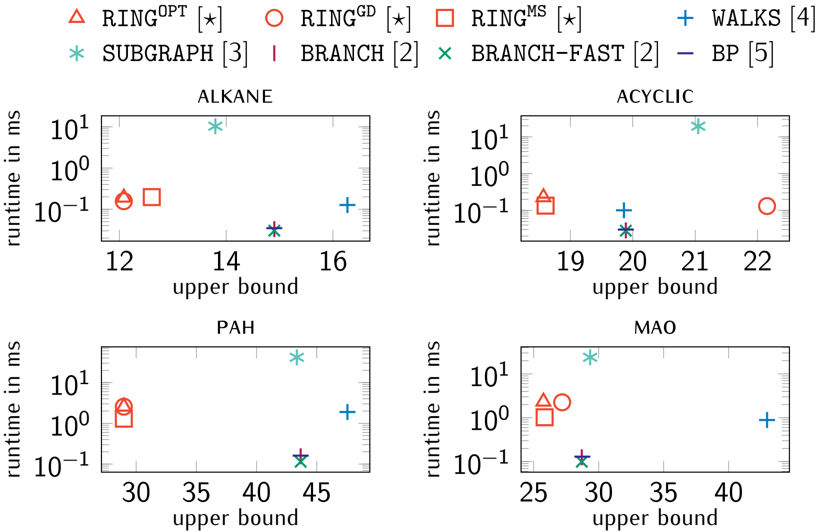
$l \leftarrow 0$; $V \leftarrow \emptyset$; $OE \leftarrow \emptyset$; $IE \leftarrow \emptyset$; $\mathcal{R}^G_L(u) \leftarrow ((\emptyset,\emptyset,\emptyset)_l)^{L-1}_{l=0}$; *// initialize ring*
$\mathtt{d}[u] \leftarrow 0$; **for** $u' \in V^G \setminus \{u\}$ **do** $\mathtt{d}[u'] \leftarrow \infty$; *// initialize distances to root*
for $e \in E^G$ **do** $\mathtt{discovered}[e] \leftarrow \mathtt{false}$; *// mark all edges as undiscovered*
$\mathtt{open} \leftarrow \{u\}$; *// initialize FIFO queue*
while $\mathtt{open} \neq \emptyset$ **do** *// main loop*
  $u' \leftarrow \mathtt{open.pop}()$; *// pop node from queue*
  if $\mathtt{d}[u'] > l$ **then** *// the l^{th} layer is complete*
    $\mathcal{R}^G_L(u)_l = (V, OE, IE)$; $l \leftarrow l+1$; *// store l^{th} layer and increment l*
    $V \leftarrow \emptyset$; $OE \leftarrow \emptyset$; $IE \leftarrow \emptyset$; *// reset nodes, inner, and outer edges*
  $V \leftarrow V \cup \{u'\}$; *// u' is node at l^{th} layer*
  for $u'u'' \in E^G$ **do** *// iterate through neighbours of u'*
    if $\mathtt{discovered}[u'u'']$ **then continue**; *// skip discovered edges*
    if $\mathtt{d}[u''] = \infty$ **then** *// found new node*
      $\mathtt{d}[u''] \leftarrow l+1$; *// set distance of new node*
      if $\mathtt{d}[u''] < L$ **then** $\mathtt{open.push}(u'')$; *// add close new node to queue*
    if $\mathtt{d}[u''] = l$ **then** $IE \leftarrow IE \cup \{u'u''\}$; *// $u'u''$ is inner edge at l^{th} layer*
    else $OE \leftarrow OE \cup \{u'u''\}$; *// $u'u''$ is outer edge at l^{th} layer*
    $\mathtt{discovered}[u'u''] \leftarrow \mathtt{true}$; *// mark $u'u''$ as discovered*
$\mathcal{R}^G_L(u)_l = (V, OE, IE)$; **return** $\mathcal{R}^G_L(u)$; *// store last layer and return ring*

Experiments

Setup

- datasets: ALKANE, ACYCLIC, PAH, MAO from GREYC's Chemistry Dataset (`https://brunl01.users.greyc.fr/CHEMISTRY/`): contain graphs representing chemical compounds
- edit costs: edit costs 1 defined in [1]
- compared methods:
  - $\texttt{RING}^{\texttt{OPT}}$: rings using optimal LSAPE for defining ϕ_V and ϕ_E
  - $\texttt{RING}^{\texttt{GD}}$: rings using greedy LSAPE for defining ϕ_V and ϕ_E
  - $\texttt{RING}^{\texttt{MS}}$: rings using multisets for defining ϕ_V and ϕ_E
  - BP: LSAPE based method suggested in [5]
  - BRANCH: LSAPE based method suggested in [2]
  - BRANCH-FAST: LSAPE based method suggested in [2]
  - WALKS: LSAPE based method suggested in [4]
  - SUBGRAPH: LSAPE based method suggested in [3]

Results


RING^{OPT} [⋆]
RING^{GD} [⋆]
RING^{MS} [⋆]
WALKS [4]
SUBGRAPH [3]
BRANCH [2]
BRANCH-FAST [2]
BP [5]
ALKANE
ACYCLIC
PAH
MAO
runtime in ms
upper bound
10^{-1}
10^{0}
10^{1}
12
14
16
19
20
21
22
30
35
40
45
25

References

[1] Z. Abu-Aisheh, B. Gaüzere, S. Bougleux, J.-Y. Ramel, L. Brun, R. Raveaux, P. Héroux, and S. Adam. "Graph edit distance contest 2016: Results and future challenges". In: *Pattern Recogn. Lett.* 100 (2017), pp. 96–103.

[2] D. B. Blumenthal and J. Gamper. "Improved Lower Bounds for Graph Edit Distance". In: *IEEE Trans. Knowl. Data Eng.* 30.3 (2018), pp. 503–516.

[3] V. Carletti, B. Gaüzère, L. Brun, and M. Vento. "Approximate Graph Edit Distance Computation Combining Bipartite Matching and Exact Neighborhood Substructure Distance". In: *GbRPR 2015*. Ed. by C. Liu, B. Luo, W. G. Kropatsch, and J. Cheng. Vol. 9069. LNCS. Cham: Springer, 2015, pp. 188–197.

[4] B. Gaüzère, S. Bougleux, K. Riesen, and L. Brun. "Approximate Graph Edit Distance Guided by Bipartite Matching of Bags of Walks". In: *S+SSPR 2014*. Ed. by P. Fränti, G. Brown, M. Loog, F. Escolano, and M. Pelillo. Vol. 8621. LNCS. Cham: Springer, 2014, pp. 73–82.

[5] K. Riesen and H. Bunke. "Approximate Graph Edit Distance Computation by Means of Bipartite Graph Matching". In: *Image Vis. Comput.* 27.7 (2009), pp. 950–959.