Graph Transformation in Constant Time

Mike Dodds & Detlef Plump

University of York

Sales Pitch

Graph transformation is expensive: finding a match for a left-hand side L in graph G requires time $O(size(G)^{size(L)})$

It becomes much easier if we can identify *uniquely-labelled nodes* ('roots') in rules and host graphs: matching requires time $O(size(L))$ if the outdegree of nodes is bounded. This is *constant* if L is considered fixed.

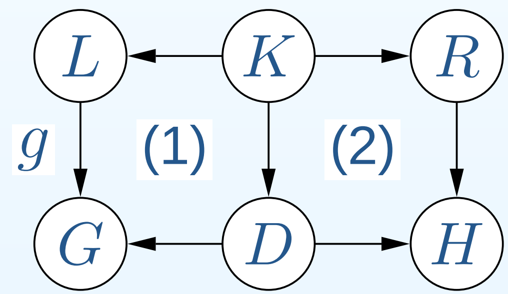
Rooted graph-transformation is surprisingly powerful, despite restrictions.

Graph reduction systems with rooted rules can recognise certain non-context-free graph languages in linear time.

The Double-Pushout Approach

A *rule* $r = \langle L \leftarrow K \rightarrow R \rangle$ consists of two inclusions $K \rightarrow L$ and $K \rightarrow R$ over graphs with possibly unlabelled nodes (where unlabelled nodes satisfy certain conditions).

A direct derivation $G \Rightarrow_{r,g} H$ consists of two natural pushouts as follows:



. . . where $g : L \rightarrow G$ is injective.

Given r and g, $G \Rightarrow_{r,g} H$ exists iff g satisfies the *dangling condition*: no node in $g(L) - g(K)$ must be incident to an edge in $G - g(L)$

Graph Transformation Problem (GTP)

Given: A graph class $\mathbb{C}$ and a $\mathbb{C}$-preserving rule r.

Input: A graph G in $\mathbb{C}$.

Output: The set $\{H \mid G \Rightarrow_r H\}$.

Cost of GTP for $r = \langle L \leftarrow K \rightarrow R \rangle$ is dominated by the cost of finding injections $g\colon L \rightarrow G$.

Graph Matching Problem (GMP):

Given: A graph class $\mathbb{C}$ and a $\mathbb{C}$-preserving rule r.

Input: A graph G in $\mathbb{C}$.

Output: The set $\{g\colon L \rightarrow G \mid g \text{ is injective}\}$.

Time complexity for GMP: $O(|G|^{|L|})$

Rooted Graph Transformation

$r = \langle L \leftarrow K \rightarrow R \rangle$ and graph class $\mathbb{C}$ conform to *Condition I* if there exists root label $\varrho \in \mathcal{C}_V$ and a bound $b \geq 0$ and s.t:

- L contains a unique ϱ-labelled node from which every node is reachable.
- For every graph G in $\mathbb{C}$:
  - exactly one node in G is labelled with the root label ϱ
  - the out-degree of every node in G bounded by b

Edge Enumeration

An *edge enumeration* of a rooted rule $r = \langle L \leftarrow K \rightarrow R \rangle$ is a sequence of edges $e_1, \ldots, e_n$ such that:

- $E_L = \{e_1, \ldots, e_n\}$.
- For each e_i either
  1. the source of e_i is labelled with the root label ϱ or
  2. there exists $j < i$ such that the target of e_j is the source of e_i.

Note: By Condition I there exists an edge enumeration for r.

Graph Matching Algorithm

The algorithm solves the GMP for a rule r and graph $G \in \mathbb{C}$ conforming to Condition I. It assumes an edge enumeration $e_1, \ldots, e_n$ of r.

$A_0 \Leftarrow$ partial injection matching only the root node
for $i = 1$ to n **do**
  if target of e_i has not been matched **then**
    $A_i \Leftarrow$ partial injections extending those in
        A_{i-1} by e_i and its target node
  else
    $A_i \Leftarrow$ partial injections extending those in A_{i-1} by e_i

Correctess follows inductively from the fact that each iteration i finds all partial injections matching edges $e_1, \ldots, e_i$.

Complexity of GMP Under Condition I

Theorem: Algorithm terminates in time $\Sigma_{i=0}^{n} b^i$ under Condition I.

Proof: To construct A_i, each iteration i extends the partial injections in A_{i-1} with edge e_i. By the out-degree bound in Condition I, each morphism can be extended in at most b ways, so each iteration takes at worst time $b|A_{i-1}|$.

Recursively expanding the sum over all iterations gives an upper time bound of:

$$1 + b + bb + \ldots + b^n = \sum_{i=0}^{n} b^i.$$

Also, the maximum size of the resulting set A_n is b^n.

Solving the GTP under Condition I

Corollary: The GTP can be solved in time $\sum_{i=0}^{n} +4|r|b^n$

Proof: Given an injection $g : L \rightarrow G$, rule application consists of checking the dangling condition, and adding, relabelling and deleting nodes. For rule r this can be completed in time $4|r|$, where $|r| = max(|L|, |R|)$.

The graph matching algorithm generates at most b^n injections, giving an upper time bound of

$$\sum_{i=0}^{n} b^i + 4|r|b^n$$

Under the GMP and Condition I, r and b are fixed, so the time complexity is constant.

Condition II

$r = \langle L \leftarrow K \rightarrow R \rangle$ and graph class $\mathbb{C}$ conform to *Condition II* if there exists a root label ϱ such that:

- L contains a unique ϱ-labelled node from which each node is reachable.
- For every graph G in $\mathbb{C}$:
  - exactly one node in G is labelled with the root label ϱ
  - distinct edges outgoing from the same node have distinct labels

Note: Condition II implies Condition I: choose b as the size of $\mathcal{C}_E$. The converse does not hold in general.

Complexity of GMP and GTP under Condition II

Theorem: Under Condition II, the graph matching algorithm requires time $n|\mathcal{C}_E| + 1$, and solving the GTP requires time $n|\mathcal{C}_E| + 1 + 4|r|$.

Proof: As edges from a node must be distinctly labelled, at iteration i each morphism in set A_{i-1} can be extended in only one way. Expanding as before gives:

$$1 + |\mathcal{C}_E| + \ldots + |\mathcal{C}_E| = n|\mathcal{C}_E| + 1$$

The time bound for the GTP follows as before.

Recognition of Graph Languages

A *signature* $\Sigma = \langle \mathcal{C}, \varrho, type \rangle$ consists of a label alphabet $\mathcal{C}$, root label ϱ, and mapping $type\colon \mathcal{C}_V \to 2^{\mathcal{C}_E}$. A graph is a Σ-*graph* if

- it contains a unique ϱ-labelled root,
- nodes have only out-edges permitted by the $type$ mapping,
- distinct out-edges have distinct labels.

Graph Reduction Specification (GRS)

GRS $\langle \Sigma, \mathcal{R}, Acc \rangle$:

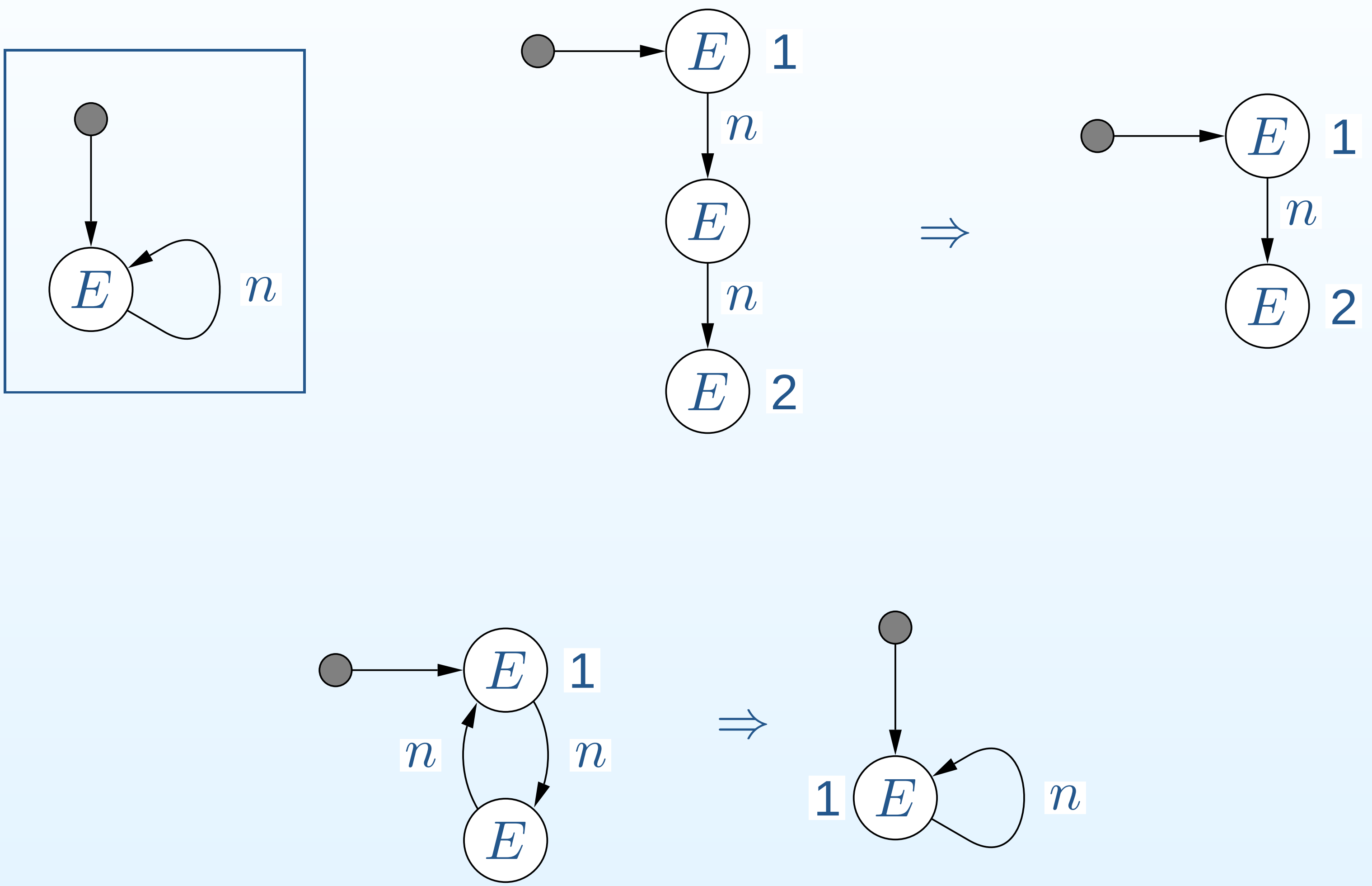
- Σ is a graph signature,
- $\mathcal{R}$ is a finite set of rooted Σ-graph preserving reduction rules, and
- Acc, a Σ-graph, is the *accepting* graph for the reduction system

A GRS *recognises* the language $L = \{G \mid G \Rightarrow^*_{\mathcal{R}} Acc\}$ of Σ-graphs.

Extension with a set $\mathcal{C}_N$ of *nonterminal* labels:
$L = \{G \mid G \Rightarrow^*_{\mathcal{R}} Acc\}$ and G is terminally labelled.

Note: Σ-graphs and GRSs conform to Condition II

Example: GRS for Cyclic Lists



Cyclic List GRS: Soundness and Completeness

Soundness (Only cyclic lists are in L):

- Show this by deriving *from* Acc using inverse rules.
- The cyclic list property is invariant for all r^{-1} such that $r \in \mathcal{R}$.

Completeness (All cyclic lists are in L):

- Acc, the smallest cyclic list, is a member of L.
- Every larger cyclic list can be reduced by some rule $r \in \mathcal{R}$ to give a smaller cyclic list.
- Hence by induction every cyclic list is reducible to Acc.

Cyclic List GRS: Termination and Closedness

Closedness ($\mathcal{R}$ preserves cyclic lists):

- For every cyclic list G, $G \Rightarrow_{\mathcal{R}} H$ implies that H is a cyclic list
- Hence membership of L can be decided without backtracking
- Procedure: Apply reduction rules as long as possible and check if the result is Acc.

Termination:

- All rules are size-reducing, so termination will occur in at most $|G|$ steps, where G is the reduced graph.

Linear GRSs

A *linear* GRS is linearly terminating and closed:

- A GRS $\langle \Sigma, \mathcal{C}_N, \mathcal{R}, Acc \rangle$ is *linearly terminating* if there is a natural number c such that for every derivation $G \Rightarrow G_1 \Rightarrow \ldots \Rightarrow G_n$ on Σ-graphs, $n \leq c|G|$.
- It is *closed* if for every step $G \Rightarrow H$ on Σ-graphs, $G \Rightarrow^* Acc$ implies $H \Rightarrow^* Acc$.

Example: The cyclic list GRS is linear.

Recognition of Graph Languages in Linear Time

The *recognition problem* for GRS languages:

Given: A GRS $S = \langle \Sigma, \mathcal{C}_N, \mathcal{R}, Acc \rangle$.

Input: A Σ-graph G.

Output: Does G belong to $\mathrm{L}(S)$?

Theorem: For linear GRSs, the recognition problem can be decided in linear time.

Proof: Membership in $L(S)$ is tested for a (terminally labelled) Σ-graph G as follows: Apply as long as possible and check that the resulting graph is Acc. This is correct by closedness. By linear termination and constant time complexity of GTP under Condition II, the time needed is linear in $|G|$.

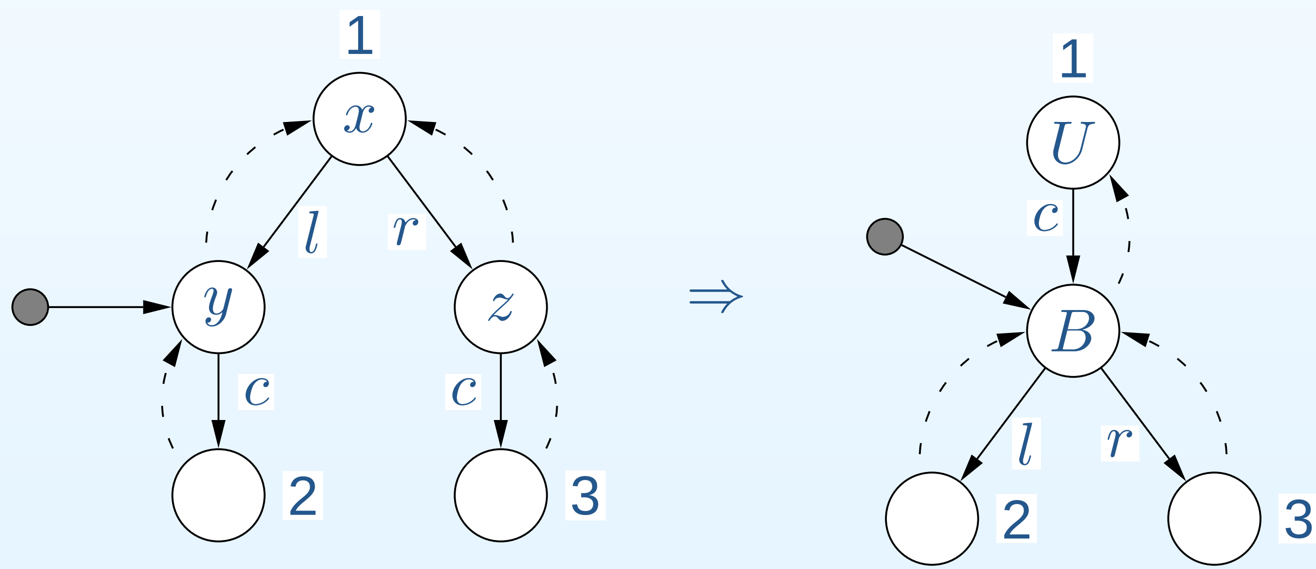
Example: Balanced Binary Trees with Back-pointers

Binary trees such that all paths from the tree-root to a leaf are of the same length. Back-pointers are needed: Condition I / II requires that all nodes are reachable from a ϱ-root.

Sample rule:

R3(l):

$x \in \{B, B'\}$
$y \in \{U, U'\}$
$z \in \{U, U'\}$



BBTB GRS Properties

Proposition: GRS BB given in the paper is a linear GRS specifying the set of all BBTBs.

Note: The language of BBTBs is not context-free (in the sense of hyperedge replacement or node replacement grammars).

Future Work

- Modified constraints: when bounding both in- and out-degree of all nodes, nodes on left-hand sides need not be reachable from the root. This allows us to eliminate backpointers in BBTBs.
- Relation to unrooted graph transformation: translation of unrooted into rooted systems?
- Relation to work on recognising languages of bounded treewidth in linear time (e.g. by Bodlaender & de Fluiter)