

Graph Transformation

Detlef Plump

The University of York

Some roots of rule-based graph transformation

Kolmogorov, Uspenskij (53/58)
(computability theory)

Pfaltz, Rosenfeld (69)
Montanari (70)
Pfaltz (72)
Rosenfeld, Milgram (72)
(web grammars for
picture description)

Feder (71)
Mylopoulos (72)
Pavlidis (72)
(graph grammars)

Ehrig, Pfender, Schneider (73)
(double-pushout approach)

T. Pratt (69/71)
Pratt, Friedman (71)
(semantics of prog.
languages)

Vuillemin (74)
Ehrig, Rosen (76)
Padawitz (79)
Staples (80)
(term rewriting
systems)

Wadsworth (71)
Staples (79)
Lamping (90)
Kathail (90)
(λ -calculus)

D. Turner (79)
(functional prog.
languages)

Some roots of rule-based graph transformation

Kolmogorov, Uspenskij (53/58)
(computability theory)

Pfaltz, Rosenfeld (69)
Montanari (70)
Pfaltz (72)
Rosenfeld, Milgram (72)
(web grammars for
picture description)

Feder (71)
Mylopoulos (72)
Pavlidis (72)
(graph grammars)

Ehrig, Pfender, Schneider (73)
(double-pushout approach)

T. Pratt (69/71)
Pratt, Friedman (71)
(semantics of prog.
languages)

Vuillemin (74)
Ehrig, Rosen (76)
Padawitz (79)
Staples (80)
(term rewriting
systems)

Wadsworth (71)
Staples (79)
Lamping (90)
Kathail (90)
(λ -calculus)

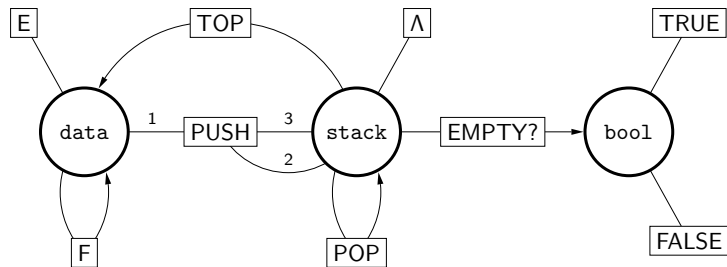
D. Turner (79)
(functional prog.
languages)

Graphs come in many flavours

directed/undirected, labelled/unlabelled, attributed, partial,
hypergraphs, hierarchical graphs, ...

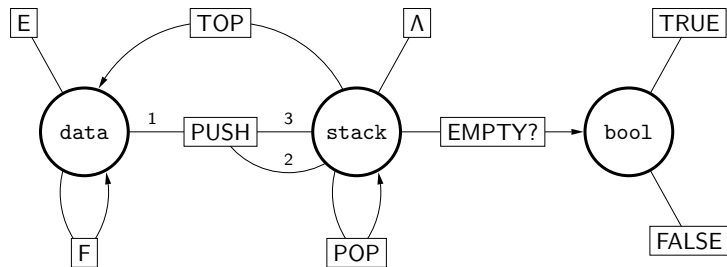
Graphs come in many flavours

directed/undirected, labelled/unlabelled, attributed, partial, hypergraphs, hierarchical graphs, ...



Graphs come in many flavours

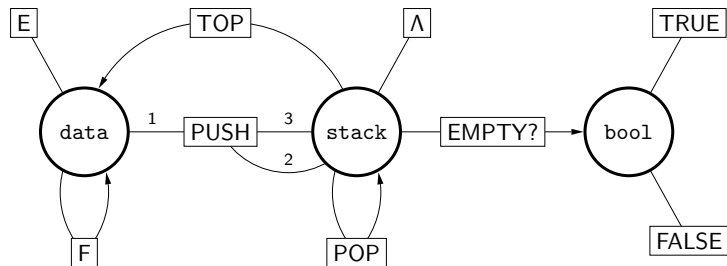
directed/undirected, labelled/unlabelled, attributed, partial, hypergraphs, hierarchical graphs, ...



- ▶ **Hypergraph** $G = \langle V, E, \text{mark}, \text{lab}, \text{att} \rangle$:
 V, E : finite sets of vertices (nodes) and hyperedges;
 $\text{mark}: V \rightarrow \Sigma_V$; $\text{lab}: E \rightarrow \Sigma_E$; $\text{att}: E \rightarrow V^*$

Graphs come in many flavours

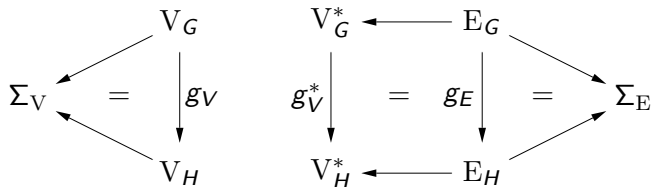
directed/undirected, labelled/unlabelled, attributed, partial, hypergraphs, hierarchical graphs, ...



- ▶ **Hypergraph** $G = \langle V, E, \text{mark}, \text{lab}, \text{att} \rangle$:
 V, E : finite sets of vertices (nodes) and hyperedges;
 $\text{mark}: V \rightarrow \Sigma_V$; $\text{lab}: E \rightarrow \Sigma_E$; $\text{att}: E \rightarrow V^*$
- ▶ G is a **graph** if $|\text{att}(e)| = 2$ for all $e \in E$

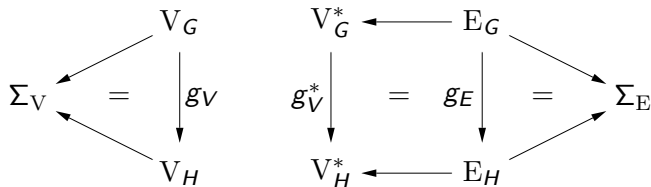
Graph morphisms

- ▶ *Graph morphism* $g: G \rightarrow H$ consists of functions $g_V: V_G \rightarrow V_H$ and $g_E: E_G \rightarrow E_H$ such that



Graph morphisms

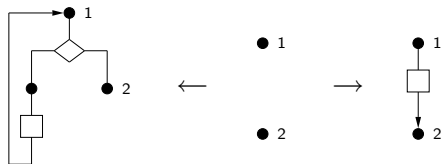
- ▶ *Graph morphism* $g: G \rightarrow H$ consists of functions $g_V: V_G \rightarrow V_H$ and $g_E: E_G \rightarrow E_H$ such that



- ▶ G and H are *isomorphic*, $G \cong H$, if g_V and g_E are bijections

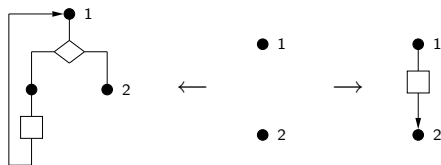
Rules (injective case)

- *Rule* $r = \langle L \leftarrow K \rightarrow R \rangle$, where $K \rightarrow L$, $K \rightarrow R$ are inclusions



Rules (injective case)

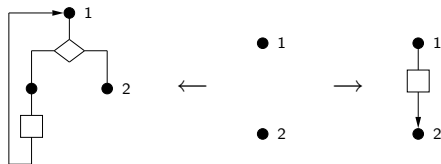
- ▶ *Rule* $r = \langle L \leftarrow K \rightarrow R \rangle$, where $K \rightarrow L$, $K \rightarrow R$ are inclusions



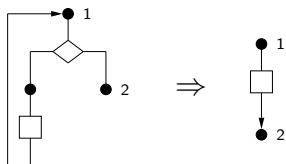
- ▶ Intuition: Remove $L - K$ and add $R - K$

Rules (injective case)

- ▶ **Rule** $r = \langle L \leftarrow K \rightarrow R \rangle$, where $K \rightarrow L$, $K \rightarrow R$ are inclusions

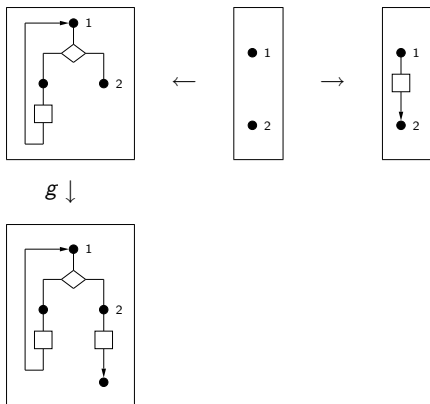


- ▶ Intuition: Remove $L - K$ and add $R - K$
- ▶ Shorthand notation:



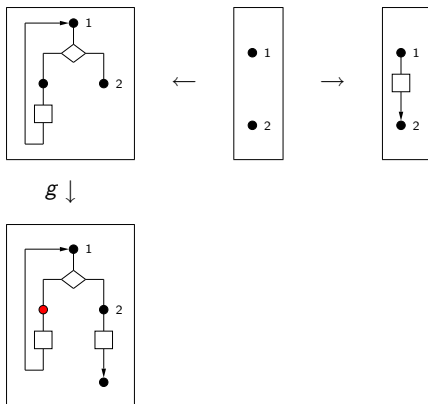
Transformation step $G \Rightarrow H$: operational description

Given: rule $\langle L \leftarrow K \rightarrow R \rangle$ and injective $g: L \rightarrow G$



Transformation step $G \Rightarrow H$: operational description

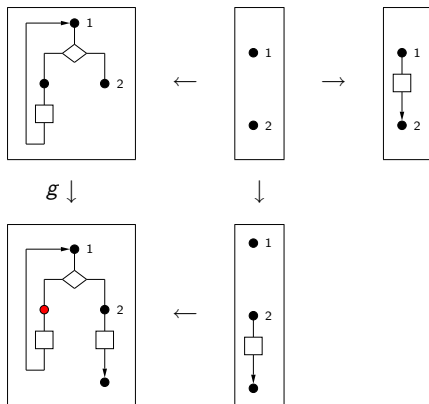
Given: rule $\langle L \leftarrow K \rightarrow R \rangle$ and injective $g: L \rightarrow G$



1. Check the *dangling condition*: no node in $g(L) - g(K)$ is incident to an edge in $G - g(L)$

Transformation step $G \Rightarrow H$: operational description

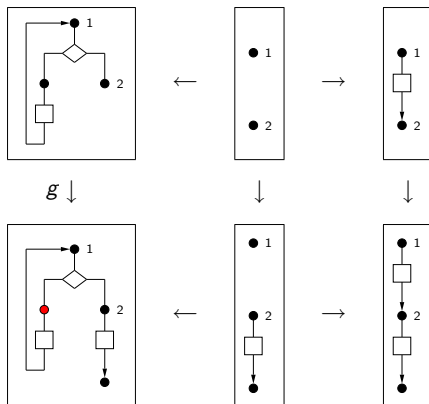
Given: rule $\langle L \leftarrow K \rightarrow R \rangle$ and injective $g: L \rightarrow G$



1. Check the *dangling condition*: no node in $g(L) - g(K)$ is incident to an edge in $G - g(L)$
2. Remove $g(L) - g(K)$

Transformation step $G \Rightarrow H$: operational description

Given: rule $\langle L \leftarrow K \rightarrow R \rangle$ and injective $g: L \rightarrow G$

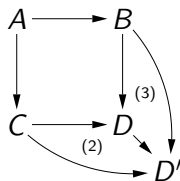
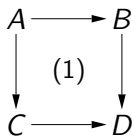


1. Check the *dangling condition*: no node in $g(L) - g(K)$ is incident to an edge in $G - g(L)$
2. Remove $g(L) - g(K)$
3. Add $R - K$

Pushouts

Diagram (1) is a *pushout* if it commutes and the following *universal property* holds:

For all morphisms $B \rightarrow D'$ and $C \rightarrow D'$ such that $A \rightarrow B \rightarrow D' = A \rightarrow C \rightarrow D'$, there is a unique morphism $D \rightarrow D'$ such that diagrams (2) and (3) commute.



Intuition: Glue together B and C in the common part A .

Transformation step $G \Rightarrow H$: pushout description

Theorem

Let $r = \langle L \leftarrow K \rightarrow R \rangle$ be a rule and $g: L \rightarrow G$ an injective morphism. Then $G \Rightarrow_{r,g} H$ if and only if there exist two pushouts of the following form:

$$\begin{array}{ccccc} L & \longleftarrow & K & \longrightarrow & R \\ \downarrow g & & \downarrow & & \downarrow \\ G & \longleftarrow & D & \longrightarrow & H \end{array}$$

Moreover, r and g determine D and H uniquely up to isomorphism.

Graph grammars

- ▶ *Graph grammar* $GG = \langle \Sigma, \mathcal{R}, S \rangle$:
 - ▶ $\Sigma = \langle \Sigma_V, \Sigma_E \rangle$: sets of node labels and edge labels
 - ▶ \mathcal{R} : finite set of rules
 - ▶ S : start graph

Graph grammars

- ▶ *Graph grammar* $GG = \langle \Sigma, \mathcal{R}, S \rangle$:
 - ▶ $\Sigma = \langle \Sigma_V, \Sigma_E \rangle$: sets of node labels and edge labels
 - ▶ \mathcal{R} : finite set of rules
 - ▶ S : start graph
- ▶ *Graph language* specified by GG : $L(GG) = \{G \mid S \Rightarrow_{\mathcal{R}}^* G\}$

Graph grammars

- ▶ *Graph grammar* $GG = \langle \Sigma, \mathcal{R}, S \rangle$:
 - ▶ $\Sigma = \langle \Sigma_V, \Sigma_E \rangle$: sets of node labels and edge labels
 - ▶ \mathcal{R} : finite set of rules
 - ▶ S : start graph
- ▶ *Graph language* specified by GG : $L(GG) = \{G \mid S \Rightarrow_{\mathcal{R}}^* G\}$
- ▶ Usually extended to allow nonterminal labels

Graph grammars

- ▶ *Graph grammar* $GG = \langle \Sigma, \mathcal{R}, S \rangle$:
 - ▶ $\Sigma = \langle \Sigma_V, \Sigma_E \rangle$: sets of node labels and edge labels
 - ▶ \mathcal{R} : finite set of rules
 - ▶ S : start graph
- ▶ *Graph language* specified by GG : $L(GG) = \{G \mid S \Rightarrow_{\mathcal{R}}^* G\}$
- ▶ Usually extended to allow nonterminal labels
- ▶ GG is *context-free* (or a *hyperedge replacement grammar*) if for each rule $\langle L \leftarrow K \rightarrow R \rangle$ in \mathcal{R} , L is a single hyperedge with its attachment nodes and K is obtained from L by removing the hyperedge

Example: grammar for control-flow graphs (1)

- ▶ Generates a subset of the “semi-structured flow graphs” of Farrow, Kennedy and Zucconi [FOCS 76]

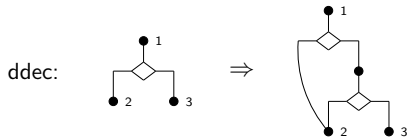
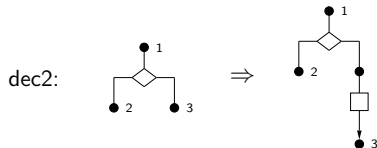
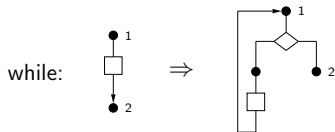
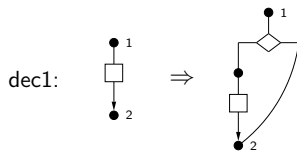
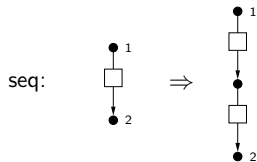
Example: grammar for control-flow graphs (1)

- ▶ Generates a subset of the “semi-structured flow graphs” of Farrow, Kennedy and Zucconi [FOCS 76]
- ▶ Start graph:



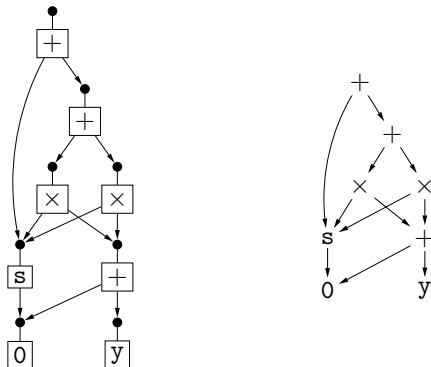
- ▶ Rules: ...

Example: grammar for control-flow graphs (2)



Term graph rewriting (1)

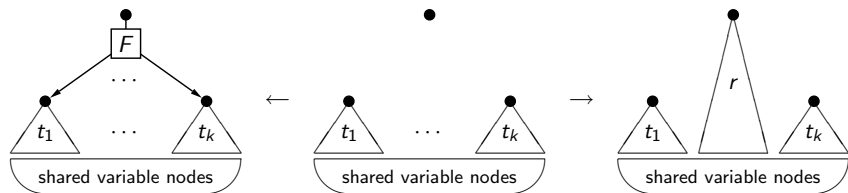
Represent terms as graphs to *share* common subexpressions



$$s(0) + ((s(0) \times (0 + y)) + (s(0) \times (0 + y)))$$

Term graph rewriting (2)

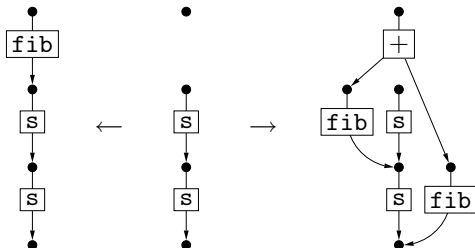
Translation of a term rewrite rule $F(t_1, \dots, t_k) \rightarrow r$:



Example: Fibonacci function

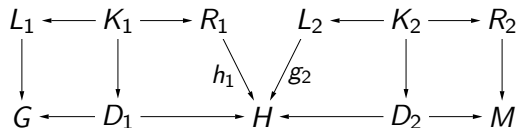
$$\begin{aligned}\text{fib}(0) &\rightarrow 0 \\ \text{fib}(s(0)) &\rightarrow s(0) \\ \text{fib}(s(s(x))) &\rightarrow \text{fib}(s(x)) + \text{fib}(x)\end{aligned}$$

Translation of recursive rule:

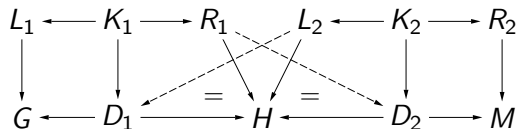


Sequential independence (1)

Steps $G \Rightarrow_r H \Rightarrow_s M$ with diagrams



are *sequentially independent* if there are morphisms as follows:



Sequential independence (1)

Steps $G \Rightarrow_r H \Rightarrow_s M$ with diagrams

$$\begin{array}{ccccccc} L_1 & \longleftarrow & K_1 & \longrightarrow & R_1 & & L_2 & \longleftarrow & K_2 & \longrightarrow & R_2 \\ & & \downarrow & & \searrow^{h_1} & & \nearrow_{g_2} & & \downarrow & & \downarrow \\ & & G & \longleftarrow & D_1 & \longrightarrow & H & \longleftarrow & D_2 & \longrightarrow & M \end{array}$$

are *sequentially independent* if there are morphisms as follows:

$$\begin{array}{ccccccc} L_1 & \longleftarrow & K_1 & \longrightarrow & R_1 & & L_2 & \longleftarrow & K_2 & \longrightarrow & R_2 \\ & & \downarrow & & \searrow & & \nearrow & & \downarrow & & \downarrow \\ & & G & \longleftarrow & D_1 & \longrightarrow & H & \longleftarrow & D_2 & \longrightarrow & M \end{array}$$

\equiv \equiv

Lemma

$G \Rightarrow_r H \Rightarrow_s M$ are sequentially independent if and only if $h_1(R_1) \cap g_2(L_2) \subseteq h_1(K_1) \cap g_2(K_2)$.

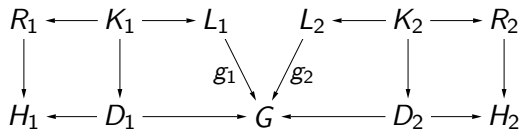
Sequential independence (2)

Theorem

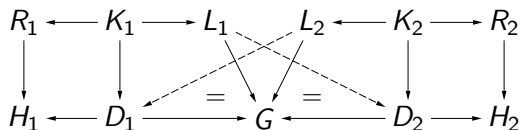
If $G \Rightarrow_r H \Rightarrow_s M$ are sequentially independent, then there are steps $G \Rightarrow_s H' \Rightarrow_r M$.

Parallel independence

Steps $H_1 \leftarrow_r G \Rightarrow_s H_2$ with diagrams



are *parallelly independent* if there are morphisms as follows:



Parallel independence

Steps $H_1 \leftarrow_r G \Rightarrow_s H_2$ with diagrams

$$\begin{array}{ccccccc} R_1 & \longleftarrow & K_1 & \longrightarrow & L_1 & & L_2 & \longleftarrow & K_2 & \longrightarrow & R_2 \\ & & \downarrow & & \searrow & & \swarrow & & \downarrow & & \downarrow \\ & & H_1 & \longleftarrow & D_1 & \longrightarrow & G & \longleftarrow & D_2 & \longrightarrow & H_2 \end{array}$$

g_1 g_2

are *parallelly independent* if there are morphisms as follows:

$$\begin{array}{ccccccc} R_1 & \longleftarrow & K_1 & \longrightarrow & L_1 & & L_2 & \longleftarrow & K_2 & \longrightarrow & R_2 \\ & & \downarrow & & \searrow & & \swarrow & & \downarrow & & \downarrow \\ & & H_1 & \longleftarrow & D_1 & \longrightarrow & G & \longleftarrow & D_2 & \longrightarrow & H_2 \end{array}$$

\equiv \equiv

Lemma

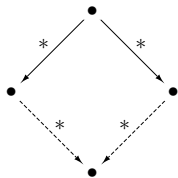
$H_1 \leftarrow_r G \Rightarrow_s H_2$ are *parallelly independent* if and only if $g_1(L_1) \cap g_2(L_2) \subseteq g_1(K_1) \cap g_2(K_2)$.

Parallel independence (2)

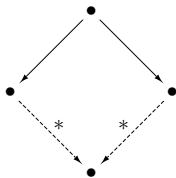
Theorem

If $H_1 \Leftarrow_r G \Rightarrow_s H_2$ are parallelly independent, then there are steps $H_1 \Rightarrow_s M \Leftarrow_r H_2$.

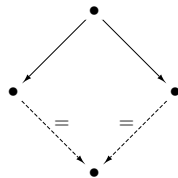
Confluence properties of binary relations



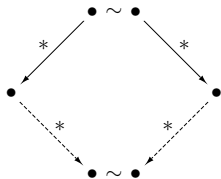
confluence



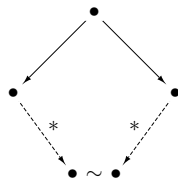
local confluence



subcommutativity



confluence modulo \sim



local confluence modulo \sim

Graph transformation systems

Graph transformation system $GT = \langle \Sigma, \mathcal{R} \rangle$

$\Sigma = \langle \Sigma_V, \Sigma_E \rangle$: sets of node labels and edge labels

\mathcal{R} : set of rules

Confluence of graph transformation

- ▶ *Graph transformation modulo isomorphism*. Define $\Rightarrow_{\mathcal{R}, \cong}$ on isomorphism classes of graphs by: $[G] \Rightarrow_{\mathcal{R}, \cong} [H]$ if there are G' and H' such that $G \cong G' \Rightarrow_{\mathcal{R}} H' \cong H$

Confluence of graph transformation

- ▶ *Graph transformation modulo isomorphism.* Define $\Rightarrow_{\mathcal{R}, \cong}$ on isomorphism classes of graphs by: $[G] \Rightarrow_{\mathcal{R}, \cong} [H]$ if there are G' and H' such that $G \cong G' \Rightarrow_{\mathcal{R}} H' \cong H$
- ▶ Note: $[G] \Rightarrow_{\mathcal{R}, \cong} [H]$ iff $G \Rightarrow_{\mathcal{R}} H$, but $[G] \Rightarrow_{\mathcal{R}, \cong}^* [H]$ need not imply $G \Rightarrow_{\mathcal{R}}^* H$

Confluence of graph transformation

- ▶ *Graph transformation modulo isomorphism*. Define $\Rightarrow_{\mathcal{R}, \cong}$ on isomorphism classes of graphs by: $[G] \Rightarrow_{\mathcal{R}, \cong} [H]$ if there are G' and H' such that $G \cong G' \Rightarrow_{\mathcal{R}} H' \cong H$
- ▶ Note: $[G] \Rightarrow_{\mathcal{R}, \cong} [H]$ iff $G \Rightarrow_{\mathcal{R}} H$, but $[G] \Rightarrow_{\mathcal{R}, \cong}^* [H]$ need not imply $G \Rightarrow_{\mathcal{R}}^* H$
- ▶ $GT = \langle \Sigma, \mathcal{R} \rangle$ is *confluent* (*locally confluent*, *subcommutative*) if $\Rightarrow_{\mathcal{R}, \cong}$ is confluent (locally confluent, subcommutative)

Confluence of graph transformation

- ▶ *Graph transformation modulo isomorphism.* Define $\Rightarrow_{\mathcal{R}, \cong}$ on isomorphism classes of graphs by: $[G] \Rightarrow_{\mathcal{R}, \cong} [H]$ if there are G' and H' such that $G \cong G' \Rightarrow_{\mathcal{R}} H' \cong H$
- ▶ Note: $[G] \Rightarrow_{\mathcal{R}, \cong} [H]$ iff $G \Rightarrow_{\mathcal{R}} H$, but $[G] \Rightarrow_{\mathcal{R}, \cong}^* [H]$ need not imply $G \Rightarrow_{\mathcal{R}}^* H$
- ▶ $GT = \langle \Sigma, \mathcal{R} \rangle$ is *confluent* (*locally confluent*, *subcommutative*) if $\Rightarrow_{\mathcal{R}, \cong}$ is confluent (locally confluent, subcommutative)

Lemma

- (1) $\langle \Sigma, \mathcal{R} \rangle$ is confluent iff $\Rightarrow_{\mathcal{R}}$ is confluent modulo isomorphism.
- (2) $\langle \Sigma, \mathcal{R} \rangle$ is locally confluent iff $\Rightarrow_{\mathcal{R}}$ is locally confluent modulo isomorphism.

Undecidability of confluence

Theorem

The following problem is undecidable in general:

Instance: A finite and terminating graph transformation system $\langle \Sigma, \mathcal{R} \rangle$ where Σ_V is a singleton.

Question: Is $\langle \Sigma, \mathcal{R} \rangle$ confluent?

Undecidability of confluence

Theorem

The following problem is undecidable in general:

Instance: A finite and terminating graph transformation system $\langle \Sigma, \mathcal{R} \rangle$ where Σ_V is a singleton.

Question: Is $\langle \Sigma, \mathcal{R} \rangle$ confluent?

- ▶ Note the difference to term and string rewriting systems, where confluence of terminating systems is decidable

Undecidability of confluence

Theorem

The following problem is undecidable in general:

Instance: A finite and terminating graph transformation system $\langle \Sigma, \mathcal{R} \rangle$ where Σ_V is a singleton.

Question: Is $\langle \Sigma, \mathcal{R} \rangle$ confluent?

- ▶ Note the difference to term and string rewriting systems, where confluence of terminating systems is decidable
- ▶ Proof by reducing the Post Correspondence Problem (PCP): every instance \mathcal{I} of the PCP is encoded as a terminating graph transformation system $\langle \Sigma(\mathcal{I}), \mathcal{R}(\mathcal{I}) \rangle$ that is confluent if and only if \mathcal{I} does not have a solution

Critical pairs

$U_1 \leftarrow_{r_1, g_1} S \Rightarrow_{r_2, g_2} U_2$ is a *critical pair* if

- (1) $S = g_1(L_1) \cup g_2(L_2)$ and
- (2) the steps are not parallelly independent

Moreover: $g_1 \neq g_2$ if $r_1 = r_2$

Critical pairs

$U_1 \leftarrow_{r_1, g_1} S \Rightarrow_{r_2, g_2} U_2$ is a *critical pair* if

- (1) $S = g_1(L_1) \cup g_2(L_2)$ and
- (2) the steps are not parallelly independent

Moreover: $g_1 \neq g_2$ if $r_1 = r_2$

Theorem

Suppose $\langle \Sigma, \mathcal{R} \rangle$ has no critical pairs. If $H_1 \leftarrow_{\mathcal{R}} G \Rightarrow_{\mathcal{R}} H_2$, then $H_1 \cong H_2$ or there are steps $H_1 \Rightarrow_{\mathcal{R}} H \leftarrow_{\mathcal{R}} H_2$.

Critical pairs

$U_1 \leftarrow_{r_1, g_1} S \Rightarrow_{r_2, g_2} U_2$ is a *critical pair* if

- (1) $S = g_1(L_1) \cup g_2(L_2)$ and
- (2) the steps are not parallelly independent

Moreover: $g_1 \neq g_2$ if $r_1 = r_2$

Theorem

Suppose $\langle \Sigma, \mathcal{R} \rangle$ has no critical pairs. If $H_1 \leftarrow_{\mathcal{R}} G \Rightarrow_{\mathcal{R}} H_2$, then $H_1 \cong H_2$ or there are steps $H_1 \Rightarrow_{\mathcal{R}} H \leftarrow_{\mathcal{R}} H_2$.

Corollary

Graph transformation systems without critical pairs are subcommutative.

Joinability

Critical pair $U_1 \leftarrow S \Rightarrow U_2$ is *joinable* if there are graphs W_1 and W_2 such that $U_1 \Rightarrow^* W_1 \cong W_2 \Leftarrow^* U_2$

Joinability

Critical pair $U_1 \leftarrow S \Rightarrow U_2$ is *joinable* if there are graphs W_1 and W_2 such that $U_1 \Rightarrow^* W_1 \cong W_2 \Leftarrow^* U_2$

Not enough for local confluence:



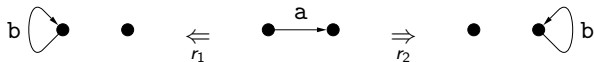
Joinability

Critical pair $U_1 \Leftarrow S \Rightarrow U_2$ is *joinable* if there are graphs W_1 and W_2 such that $U_1 \Rightarrow^* W_1 \cong W_2 \Leftarrow^* U_2$

Not enough for local confluence:



The only critical pair is joinable:



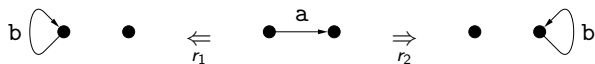
Joinability

Critical pair $U_1 \leftarrow S \Rightarrow U_2$ is *joinable* if there are graphs W_1 and W_2 such that $U_1 \Rightarrow^* W_1 \cong W_2 \Leftarrow^* U_2$

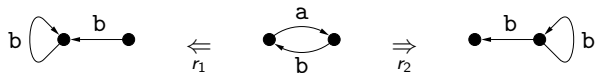
Not enough for local confluence:



The only critical pair is joinable:



But $\langle \Sigma, \{r_1, r_2\} \rangle$ is not locally confluent:



Track morphisms

- *Track morphism* $\text{tr}_{G \Rightarrow H}: G \rightarrow H$ of

$$\begin{array}{ccccc} L & \longleftarrow & K & \longrightarrow & R \\ \downarrow & & \downarrow & & \downarrow \\ G & \xleftarrow{c} & D & \xrightarrow{c'} & H \end{array}$$

is the partial graph morphism defined by

$$\text{tr}_{G \Rightarrow H}(x) = \begin{cases} c'(c^{-1}(x)) & \text{if } x \in c(D), \\ \text{undefined} & \text{otherwise} \end{cases}$$

Track morphisms

- ▶ *Track morphism* $\text{tr}_{G \Rightarrow H}: G \rightarrow H$ of

$$\begin{array}{ccccc} L & \longleftarrow & K & \longrightarrow & R \\ \downarrow & & \downarrow & & \downarrow \\ G & \xleftarrow{c} & D & \xrightarrow{c'} & H \end{array}$$

is the partial graph morphism defined by

$$\text{tr}_{G \Rightarrow H}(x) = \begin{cases} c'(c^{-1}(x)) & \text{if } x \in c(D), \\ \text{undefined} & \text{otherwise} \end{cases}$$

- ▶ Extension to derivations $G_0 \Rightarrow^* G_n$:
 id_{G_0} if $n = 0$ and $\text{tr}_{G_1 \Rightarrow^* G_n} \circ \text{tr}_{G_0 \Rightarrow G_1}$ otherwise

Strong joinability and Critical Pair Lemma

Critical pair $U_1 \Leftarrow S \Rightarrow U_2$ is *strongly joinable* if there are derivations $U_1 \Rightarrow^* W_1$, $U_2 \Rightarrow^* W_2$ and an isomorphism $i: W_1 \rightarrow W_2$ such that for each node v that is preserved by both $S \Rightarrow U_1$ and $S \Rightarrow U_2$,

- (1) $\text{tr}_{S \Rightarrow U_1 \Rightarrow^* W_1}(v)$ and $\text{tr}_{S \Rightarrow U_2 \Rightarrow^* W_2}(v)$ are defined and
- (2) $i(\text{tr}_{S \Rightarrow U_1 \Rightarrow^* W_1}(v)) = \text{tr}_{S \Rightarrow U_2 \Rightarrow^* W_2}(v)$.

Strong joinability and Critical Pair Lemma

Critical pair $U_1 \Leftarrow S \Rightarrow U_2$ is *strongly joinable* if there are derivations $U_1 \Rightarrow^* W_1$, $U_2 \Rightarrow^* W_2$ and an isomorphism $i: W_1 \rightarrow W_2$ such that for each node v that is preserved by both $S \Rightarrow U_1$ and $S \Rightarrow U_2$,

- (1) $\text{tr}_{S \Rightarrow U_1 \Rightarrow^* W_1}(v)$ and $\text{tr}_{S \Rightarrow U_2 \Rightarrow^* W_2}(v)$ are defined and
- (2) $i(\text{tr}_{S \Rightarrow U_1 \Rightarrow^* W_1}(v)) = \text{tr}_{S \Rightarrow U_2 \Rightarrow^* W_2}(v)$.

Critical Pair Lemma

A graph transformation system is locally confluent if all its critical pairs are strongly joinable.

Strong joinability and Critical Pair Lemma

Critical pair $U_1 \Leftarrow S \Rightarrow U_2$ is *strongly joinable* if there are derivations $U_1 \Rightarrow^* W_1$, $U_2 \Rightarrow^* W_2$ and an isomorphism $i: W_1 \rightarrow W_2$ such that for each node v that is preserved by both $S \Rightarrow U_1$ and $S \Rightarrow U_2$,

- (1) $\text{tr}_{S \Rightarrow U_1 \Rightarrow^* W_1}(v)$ and $\text{tr}_{S \Rightarrow U_2 \Rightarrow^* W_2}(v)$ are defined and
- (2) $i(\text{tr}_{S \Rightarrow U_1 \Rightarrow^* W_1}(v)) = \text{tr}_{S \Rightarrow U_2 \Rightarrow^* W_2}(v)$.

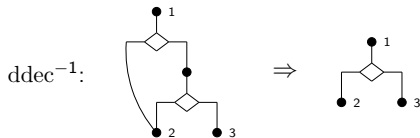
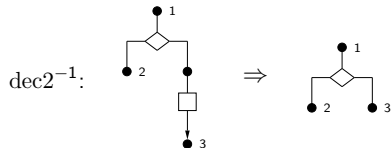
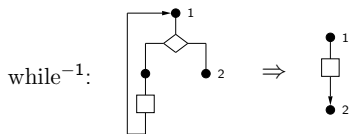
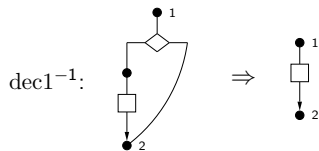
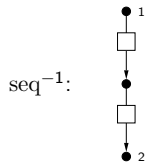
Critical Pair Lemma

A graph transformation system is locally confluent if all its critical pairs are strongly joinable.

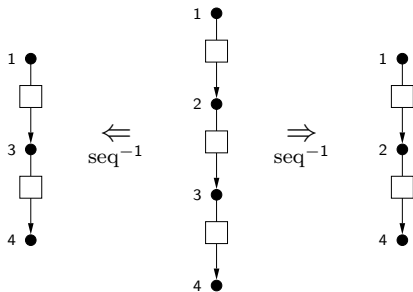
Theorem

A terminating graph transformation system is confluent if all its critical pairs are strongly joinable.

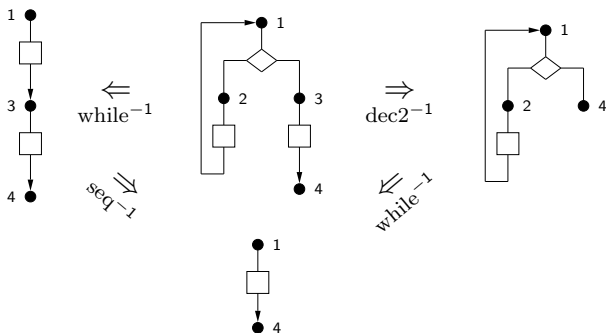
Example: reduction rules for control-flow graphs



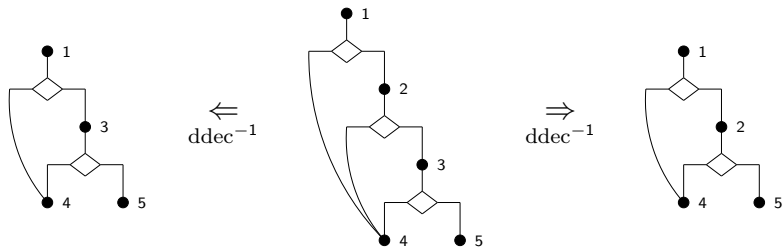
Example: critical pairs of reduction rules (1)



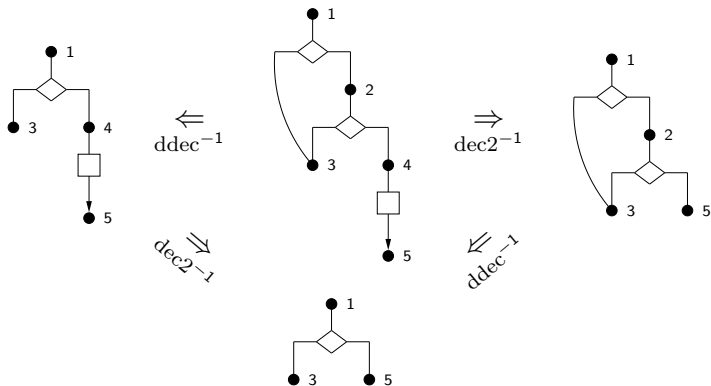
Example: critical pairs of reduction rules (2)



Example: critical pairs of reduction rules (3)

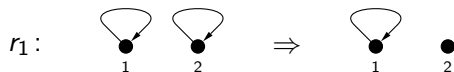


Example: critical pairs of reduction rules (4)

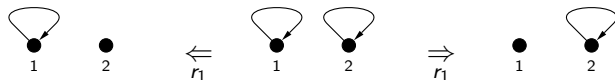


Confluence need not imply strong joinability

The system



is terminating and confluent but the critical pair



is not strongly joinable

Literature

- ▶ **Textbook**

H. Ehrig, K. Ehrig, U. Prange, G. Taentzer: Fundamentals of Algebraic Graph Transformation, Springer (2006)

- ▶ **Handbook of Graph Grammars and Computing by Graph Transformation**

Eds. G. Rozenberg, H. Ehrig, G. Engels, H.-J. Kreowski and U. Montanari, World Scientific

- ▶ Vol. 1: Foundations (1997)
- ▶ Vol. 2: Applications, Languages and Tools (1999)
- ▶ Vol. 3: Concurrency, Parallelism and Distribution (1999)

- ▶ **Term Graph Rewriting**

Chapter 1 in Vol. 2 of the Handbook