Encoding Graph Transformation in Linear Logic

Paolo Torrini

joint work with Reiko Heckel

pt95@mcs.le.ac.uk

University of Leicester

Graph Transformation

- Graph Transformation Systems (GTS) — high-level approach to system modelling, UML, model-driven development, stochastic simulation
- Existing formalisations — algebraic-categorical (SPO, DPO), 2nd-order predicate logic
- High-level character, strong mathematical foundation
- Double-pushout (DPO) — mature approach, based on category theory

Linear Logic

- Linear logic — can handle resources at the propositional level, by dropping Weakening and Contraction
- Intuitionistic variant (ILL)
- Linearity — each premise used exactly once in a deduction, each argument used once by a function
- Non-linearity recovered by means of !
- Interesting proof theory (natural deduction, sequent calculus), various implementations (declarative languages, logical frameworks)

Encoding GT in LL — why?

- LL close to process algebras (Abramsky, Pfenning, Cervesato)
- Parallel composition ($\alpha \otimes \beta$), choice ($\alpha \& \beta$), reachability ($\vdash \alpha \multimap \beta$), replication (!)
- Semantic motivation: taking closer graph transformation and process algebra
- Existing approach: hyperedge replacement
- What we do: logic-based hyperedge replacement
- Practical motivation: making proofs about GTS easier

Typed hypergraphs

- Hypergraph $G = \langle V, E, \mathsf{s} \rangle$
  V set of nodes, E set of hyperedges
  assignment $\mathsf{s} : E \to V^*$
- H-graph morphism — $\langle \phi_V : V_1 \to V_2,\ \phi_E : E_1 \to E_2 \rangle$
  assignment-preserving
- Type h-graph $TG = \langle \mathcal{V}, \mathcal{E}, \mathsf{ar} \rangle$
  $\mathcal{V}$ set of node types, $\mathcal{E}$ set of h-edge types
  $\mathsf{ar}(l) : \mathcal{E} \to \mathcal{V}^*$
- TG-typed h-graph (G, t), with $\mathsf{t} : G \to TG$
- TG-typed h-graph morphism $f : (G_1, \mathsf{t}_1) \to (G_2, \mathsf{t}_2)$
  is h-morphism $f : G_1 \to G_2$ with $\mathsf{t}_2 \circ f = \mathsf{t}_1$

DPO diagram

- Graph transformation rule $p : L \xleftarrow{l} K \xrightarrow{r} R$
  span of typed h-graph morhisms (l, r),
  K interface, L/K to be deleted, R/K to be created,
  rule application determined by match morphism m,
  m determined up to iso by interface morphism d
- DPO conditions — (1) Identification condition:
  (a) m never identifies distinct L/K elements
  (b) m never identifies L/K elements with K ones
  (2) Dangling condition: for each node $n \in L/K$, all
  edges connected to n are in L/K, too

$$
\begin{tikzcd}
L \arrow[d, "m"'] \arrow[dr, phantom, "(1)"] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, "d"] \arrow[dr, phantom, "(2)"] & R \arrow[d, "m^*"] \\
G & D \arrow[l, "g"] \arrow[r, "h"'] & H
\end{tikzcd}
$$

Graph expressions

- Algebraic characterisation of DPO-GTS:
  edge as predicates over nodes, empty graph, parallel composition,
  restriction for nodes
- Graph constituent $C = \; e(n_1, \ldots, n_k) \mid \mathsf{Nil} \mid C_1 \parallel C_2 \mid \nu n.C$
- Implicit typing — $n : A, \quad e(n_1, \ldots, n_k) : L(A_1, \ldots, A_k)$
- Graph expression $X \vDash C$
  $X \subseteq V$ is graph interface — generalisation of rule interface, includes the free nodes of C and free isolated nodes
- closed GE has empty interface

Structural congruence

- $C \equiv C'$
  - $\|$ — associative, commutative
    Nil — neutral element
  - $\nu n.\ C \equiv \nu m.\ C[m/n]$, if m does not occur free in C.
    $\nu n.\nu m.C \equiv \nu m.\nu n.C$
    $\nu n.(C_1 \parallel C_2) \equiv C_1 \parallel (\nu n.C_2)$
    if n does not occur free in C_1
- $X \vDash C \equiv Y \vDash C'$ iff $X = Y$ and $C \equiv C'$

Transformation

- $E_1 = K \vDash L$ and $E_2 = K \vDash R$
  GEs sharing no free isolated nodes
- $\Lambda\overline{x}.L \stackrel{p}{\Longrightarrow} R$ *rule expression* for $p : L \stackrel{l}{\longleftarrow} K \stackrel{r}{\longrightarrow} R$
  $\overline{x} = x_1, \ldots, x_k$ represents K as sequence of variables
- restriction to node interfaces (no edges in K)
- Application of p at match m (G closed GE),
  schema satisfies DPO conditions

$$\frac{\Lambda\overline{x}.L \stackrel{p}{\Longrightarrow} R \quad G \equiv \nu\overline{n}.L[\overline{n} \stackrel{d}{\longleftarrow} \overline{x}] \parallel C \quad H \equiv \nu\overline{n}.R[\overline{n} \stackrel{d}{\longleftarrow} \overline{x}] \parallel C}{G \stackrel{p,d}{\Longrightarrow} H} \stackrel{\langle p,m \rangle}{\Longrightarrow}$$

Overall plan

- Algebraic characterisation of DPO-GTS — hyperedge replacement-style (difference: isolated nodes)
- Translation to a quantified extension of ILL
- up to iso (typing, connectivity): edge expressions unvaried, Nil as $\mathbf{1}$, $||$ as $\otimes$, ν as $\hat{\exists}$, $\Longrightarrow$ as $\multimap$, Λ as $\forall$
- Nodes: occur as non-linear terms in edge expressions, but need linear treatment to meet DPO conditions
- full translation maps expressions to derivations, and involves proof terms (linear λ-calculus)
- terms represent identity of nodes and edges
- We translate individual graphs, then forget about terms and reason up to isomorphism

Normal forms

- (closed) h-graph as (closed) formula

$$\hat{\exists}\overline{x : A}.\gamma$$

$\overline{x : A}$ sequence of typed variables,
either $\gamma = \mathbf{1}$ or $\gamma = L_1\ (\overline{x}_1) \otimes \ldots \otimes L_k\ (\overline{x}_k)$

- Adequacy of h-graph representation
- Transformation rule as closed formula

$$\forall\overline{x : A}.\alpha \multimap \beta$$

with α, β graph formulas

Reachability

- Tansformation — G_0, G_1 closed h-graphs, G_0 initial, $P_1, \ldots, P_k$ rules
  - G_1 reachable from by some application of the rules

$$!P_1, \ldots, !P_k, G_0 \vdash G_1$$

  - G_1 reachable by applying each rule once

$$P_1, \ldots, P_k, G_0 \vdash G_1$$

- Translation complete with respect to reachability (sequent provable if graph reachable)
- Soundness — work in progress,
  general idea — logically valid implications are "read-only" transformations

QILL

- ILL extended with 1st-order quantification
- Labels attached to premises (identity of occurrences)
- Double-entry sequents — linear premises (Δ) and non-linear ones (Γ, equivalent to $!\Gamma$)

$$\Gamma = x :: (\alpha : term), \ldots, p :: (\beta : form), \ldots$$

$$\Delta = u :: (\alpha : form), \ldots$$

- Proof-terms based on linear λ-calculus
- Sequents representing derivations

$$\Gamma; \Delta \vdash N :: (\alpha : \tau)$$

Proof system — language

- $\alpha = A : term \mid L(N_1, \ldots, N_n) \mid \mathbf{1} \mid \alpha_1 \otimes \alpha_2 \mid \alpha_1 \multimap \alpha_2 \mid !\alpha_1 \mid \alpha_1 \& \alpha_2 \mid \forall x : \beta.\alpha \mid \hat{\exists} x : \beta.\alpha \mid \alpha \downharpoonleft N \mid \alpha = \alpha$
- $M = x \mid p \mid u \mid \mathsf{nil} \mid N_1 \otimes N_2 \mid \lambda x.N \mid \hat{\lambda} u.N \mid N_1 \hat{\ } N_2 \mid N_1 N_2 \mid\ !M \mid \langle N_1, N_2 \rangle \mid \mathsf{fst}\ N \mid \mathsf{snd}\ N \mid \mathsf{id}_\alpha$
- $\alpha \hat{\equiv} \beta \ =_{df}\ (\alpha \multimap \beta) \& (\beta \multimap \alpha)$
- $\alpha \#(x, N) \ =_{df}\ (\alpha[N/x])[x/N] = \alpha$
  meaning N does not occur free in $\hat{\exists} x.\alpha$
- let $P = N_1$ in $N_2 \ =_{df}\ (\lambda P.N_2) N_1$
  where P is a term pattern (does not contain abstractions)
- $\hat{\varepsilon}(N_1 | N_2).N_3 =_{df}\ N_1 \otimes !N_2 \otimes N_3$

Application schema

$$\frac{\begin{array}{l}\Gamma; \Delta \vdash \forall \overline{x : A_x}.\alpha_L \;\multimap\; \alpha_R \\ \Gamma; \cdot \vdash \alpha_G \hat{\equiv} \alpha_{G'} \\ \Gamma; \cdot \vdash \alpha_H \hat{\equiv} \alpha_{H'} \\ \alpha_{G'} = \hat{\exists} \overline{z : A_z}.\alpha_L[\overline{z : A_z} \xleftarrow{d} \overline{x : A_x}] \otimes \alpha_C \\ \alpha_{H'} = \hat{\exists} \overline{z : A_z}.\alpha_R[\overline{z : A_z} \xleftarrow{d} \overline{x : A_x}] \otimes \alpha_C\end{array}}{\Gamma; \Delta \vdash \alpha_G \multimap \alpha_H} \overset{\langle p,m \rangle}{\Longrightarrow}$$

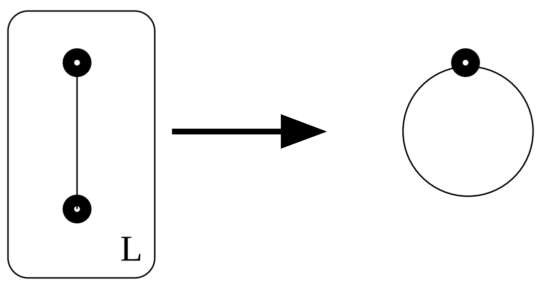
Embedding h-graphs

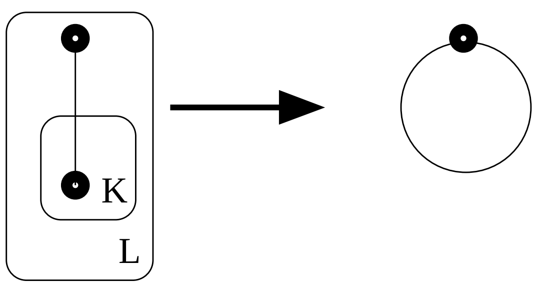
- H-graphs: edge components, empty graph Nil ($\mathbf{1}$) and parallel composition $\|$ ($\otimes$) — straightforward
- restriction ν — more problematic
- standard quantification ($\forall, \exists$) in ILL deals with non-linear terms
- at first sight — OK, nodes may have multiple occurrences in edge expressions, all we need is to handle edges linearly edge
  - we could map ν to $\exists$
    — after all, ν distributes over $\|$, $\exists$ over $\otimes$
  - not enough to meet DPO conditions

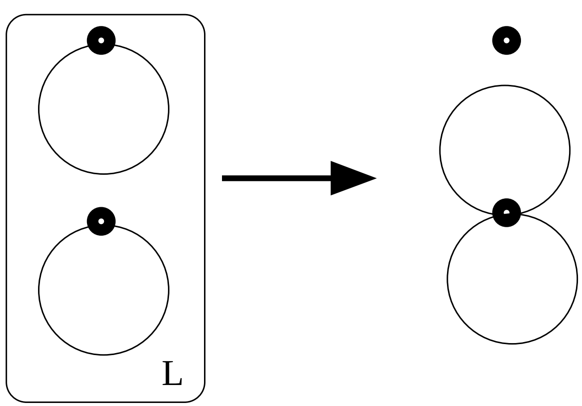
Quantifier and DPO conditions

- $\nvdash (\hat{\exists} x : \beta.\ \alpha(x,x)) \multimap \hat{\exists} xy : \beta.\ \alpha(x,y)$
the resource for x cannot suffice for x and y.
- $\nvdash \forall x : \beta.\ \beta \downharpoonleft x \otimes \alpha(x,x) \multimap \hat{\exists} y : \beta.\alpha(y,x)$
y and x should be instantiated with the same term —
blocked by the freshness condition in $\hat{\exists}$ introduction
- $\nvdash (\hat{\exists} yx : \beta.\ \alpha_1(x) \otimes \alpha_2(x)) \multimap (\hat{\exists} x : \beta.\alpha_1(x)) \otimes \hat{\exists} x : \beta.\alpha_2(x)$
the two bound variables in the consequence require
distinct resources and refer to distinct occurrences

Incorrect matches







RBQ introduction

$$\frac{\begin{array}{ll}\Gamma;\Delta \vdash M :: \alpha[N/x] & \Gamma;\cdot \vdash N :: \beta \\ \Gamma;\Delta' \vdash n :: \beta\lfloor N & \Gamma, x :: \beta;\cdot \vdash \mathrm{id}_\alpha :: (\alpha[N/x])[x/N] = \alpha\end{array}}{\Gamma;\Delta,\Delta' \vdash (!N \otimes n) \otimes M :: \hat{\exists} x : \beta.\alpha}\ \hat{\exists} I$$

- (1) $\alpha[N/x]$ graph with N in place of free x
  (2) N well-typed — enough to restrict N by x? No!
- to restrict (3) there has to be a node (linear resource) named by N — $\lfloor$ denotes lifting of type from term to formula with naming reference to term
- moreover (4) N does not occur in α (unless $N = x$) — a freshness condition, here formalised using type equality and substitution

RBQ elimination

$$\frac{\Gamma; \Delta_1 \vdash M :: \hat{\exists} x : \beta.\ \alpha \quad \Gamma, x :: \beta; \Delta_2, n :: \beta \downharpoonleft x, v :: \alpha \vdash N :: \gamma}{\Gamma; \Delta_1, \Delta_2 \vdash \text{let } (!x \otimes n) \otimes v = M \text{ in } N :: \gamma} \ \hat{\exists} E$$

- Standard elimination rule
- since we restrict only introduction, normalisation applies at least as with $\exists$
- $\hat{\exists} I$ and $\hat{\exists} E$ can be used to simulate restriction/unrestriction operationally in the logic as steps in the construction/destruction of graph expressions

$$\frac{\Gamma; \cdot \vdash N :: \alpha}{\Gamma; n :: \alpha \downharpoonleft N \vdash n :: \alpha \downharpoonleft N} \ \downharpoonleft A$$

Conclusion and further work

- Proof theory-driven approach to GT
- uses resource logic
- new quantifier to deal with restriction
- two-level embedding approach
- Interest in mechanised theorem proving
- Extension to generalised interfaces
- Stochastic GTS

rules I

$$\frac{}{\Gamma; u :: \alpha \vdash u :: \alpha}\ Id \qquad \frac{}{\Gamma, x :: \alpha; \cdot \vdash x :: \alpha}\ UId$$

$$\frac{}{\Gamma, p :: \alpha; \cdot \vdash p :: \alpha}\ FId \qquad \frac{}{\Gamma; \cdot \vdash \mathsf{id}_\alpha :: \alpha = \alpha}\ Eq$$

$$\frac{\Gamma; \Delta_1 \vdash M :: \alpha \quad \Gamma; \Delta_2 \vdash N :: \beta}{\Gamma; \Delta_1, \Delta_2 \vdash M \otimes N :: \alpha \otimes \beta}\ \otimes I$$

$$\frac{\Gamma; \Delta_1 \vdash M :: \alpha \otimes \beta \quad \Gamma; \Delta_2, u :: \alpha, v :: \beta \vdash N :: \gamma}{\Gamma; \Delta_1, \Delta_2 \vdash \mathsf{let}\ u \otimes v = M\ \mathsf{in}\ N :: \gamma}\ \otimes E$$

$$\frac{\Gamma; \Delta, u :: \alpha \vdash M :: \beta}{\Gamma; \Delta \vdash \hat{\lambda} u : \alpha .\ M :: \alpha \multimap \beta}\ \multimap I$$

$$\frac{\Gamma; \Delta_1 \vdash M :: \alpha \multimap \beta \quad \Gamma; \Delta_2 \vdash N :: \alpha}{\Gamma; \Delta_1, \Delta_2 \vdash M\hat{\ }N :: \beta}\ \multimap E$$

rules II

$$\frac{}{\Gamma; \cdot \vdash \textsf{nil} :: \mathbf{1}}\ \mathbf{1}I \qquad \frac{\Gamma; \Delta \vdash M :: \mathbf{1} \quad \Gamma; \Delta' \vdash N :: \alpha}{\Gamma; \Delta, \Delta' \vdash \textsf{let nil} = M \textsf{ in } N :: \alpha}\ \mathbf{1}E$$

$$\frac{\Gamma; \Delta \vdash M :: \alpha \quad \Gamma; \Delta \vdash N :: \beta}{\Gamma; \Delta \vdash \langle M, N \rangle :: \alpha \& \beta}\ \& I$$

$$\frac{\Gamma; \Delta \vdash M :: \alpha \& \beta}{\Gamma; \Delta \vdash \textsf{fst } M :: \alpha}\ \& E1 \qquad \frac{\Gamma; \Delta \vdash M :: \alpha \& \beta}{\Gamma; \Delta \vdash \textsf{snd } M :: \beta}\ \& E2$$

$$\frac{\Gamma; \cdot \vdash M :: \alpha}{\Gamma; \cdot \vdash !M :: !\alpha}\ !I \qquad \frac{\Gamma; \Delta_1 \vdash M :: !\alpha \quad \Gamma, p :: \alpha; \Delta_2 \vdash N :: \beta}{\Gamma; \Delta_1, \Delta_2 \vdash \textsf{let } p = M \textsf{ in } N :: \beta}\ !E$$

$$\frac{\Gamma, x :: \beta; \Delta \vdash M :: \alpha}{\Gamma; \Delta \vdash \lambda x.\ M :: \forall x : \beta.\ \alpha}\ \forall I \qquad \frac{\Gamma; \Delta \vdash M :: \forall x : \beta.\ \alpha \quad \Gamma; \cdot \vdash N :: \beta}{\Gamma; \Delta \vdash MN :: \alpha[N/x]}\ \forall E$$

Translation — I

Constituents

$$[\![e_i(m,\ldots,n) : L_i(A_m,\ldots,A_n)]\!] \;=_{df}\; Id\ [\Gamma;;\quad c_i :: L_i(x_m,\ldots,x_n)]$$

$$[\![\mathsf{Nil}]\!] \;=_{df}\; \mathbf{1}I\ [\Gamma]$$

$$[\![M \parallel N]\!] \;=_{df}\; \otimes I\ [[\![M]\!];;\quad [\![N]\!]]$$

$$\begin{aligned} [\![\nu n : A.N]\!] \;=_{df}\; & \hat{\exists} I\ [[\![N]\!];; \\ & UId\ [\Gamma;;\ x_n :: A];; \\ & \qquad Id\ [\Gamma;;\ n :: A\!\downharpoonleft\! x_n];; \\ & \qquad \Gamma, y :: A; \cdot \vdash \mathsf{id} : MainType([\![N]\!])[y/x_n]\#(y, x_n)] \end{aligned}$$

Translation — II

Graph interfaces

$$
\begin{aligned}
[\![n : A]\!] &=_{df} Id\ [\Gamma, x :: A;;\quad n :: A \downharpoonleft x] \\
[\![\{n : A\}]\!] &=_{df} [\![n : A]\!] \\
[\![\{n_1 : A_1\} \cup X]\!] &=_{df} \otimes I\ [[\![\{n_1 : A_1\}]\!];;\ [\![X]\!]]
\end{aligned}
$$

Graph expressions

$$
[\![X \vDash C]\!] =_{df} \otimes I\ [[\![X]\!]_I;;\ [\![C]\!]]
$$

Graph derivations

- *graph formulas* — $\mathbf{1}, \otimes, \hat{\exists}, \downarrow$ fragment of the logic containing only primitive graph types (node and edge types)
- *graph context* — multiset of typed nodes and typed edge components.
- *graph derivation* — derivable sequent $\Gamma; \Delta \vdash N :: \gamma$, where γ is a graph formula, Δ is a graph context, Γ the environment, N a normal derivation.
- Uses only axioms and the introduction rules $\mathbf{1}I$, $\otimes I$, $\hat{\exists} I$.

Quantifier and congruence

$\hat{\exists}$ satisfies properties of renaming, exchange and distribution over $\otimes$

- $\vdash (\hat{\exists}x : \alpha.\beta(x)) \hat{\equiv} (\hat{\exists}y : \alpha.\beta(y))$
- $\vdash (\hat{\exists}xy : \alpha.\gamma) \hat{\equiv} (\hat{\exists}yx : \alpha.\gamma)$
- $\vdash (\hat{\exists}x : \alpha.\beta \otimes \gamma(x)) \hat{\equiv} (\beta \otimes \hat{\exists}x : \alpha.\gamma(x))$ (x not in α)

Equivalence between α and $\hat{\exists}x.\ \alpha$ generally fails in both directions, even when x does not occur free in α

Graphs and types — adequacy

- Isomorphism between graph expressions and graph derivations
- Isomorphism between graphs (congruence classes of graph expressions) and graph formulas modulo linear equivalence
- Curry-Howard style correspondence
- Possibility to implement hypergraphs and to reason about them

Graph transformation

- Less interested in component identity, higher-level translation, based on logic formulas
- Linear implication as transformation
- Standard quantifier for interface nodes
- Rule names as non-linear resources (unlimited application)

$$
\begin{aligned}
[\![M \Longrightarrow N]\!]^T &=_{df} [\![M]\!]^T \multimap [\![N]\!]^T \\
[\![\Lambda x : A.N]\!]^T &=_{df} \forall x : A.[\![N]\!]^T
\end{aligned}
$$

$$
[\![\pi(p)]\!] \;=_{df}\; FId\ [\Gamma;; \quad p :: \forall \overline{x : A_x}.[\![L]\!]^T \multimap [\![R]\!]^T]
$$

Completeness and soundness

- Let $\Gamma_P = \Sigma \cup [\rho | \rho = [\![\pi(p)]\!]^T, p \in P]$, then for each reachable h-graph G

$$\Gamma_P; [\![G_0]\!]^T \vdash [\![G]\!]^T$$

- Let R be a multiset of transformations, $\Delta_R = [\tau | \tau = [\![t]\!]^T, t \in R]$, then for each h-graph G reachable from G_0 by executing R

$$\Sigma; [\![G_0]\!]^T, \Delta_R \vdash [\![G]\!]^T$$

- This is for completeness
- Soundness requires more work on the interpretation of linear implication