Nested Term Graphs

Clemens Grabmayer
Department of Computer Science
VU University Amsterdam
The Netherlands
C.A.Grabmayer@vu.nl

Vincent van Oostrom
Philosophy
Utrecht University
The Netherlands
V.vanOostrom@uu.nl

We report on work in progress on ‘nested term graphs’ for formalizing higher-order terms (e.g. finite or infinite λ-terms), including those expressing recursion (e.g. terms in the λ-calculus with letrec). The idea is to represent the nested scope structure of a higher-order term by a nested structure of term graphs. Based on a signature that is partitioned into atomic and nested function symbols, we define nested term graphs both intensionally, as tree-like recursive graph specifications that associate nested symbols with usual term graphs, and extensionally, as enriched term graph structures. These definitions induce corresponding notions of bisimulation between nested term graphs. Our main result states that nested term graphs can be implemented faithfully by first-order term graphs.

Introduction

Structures such as strings, terms, and graphs frequently come equipped with additional structure. In this paper we study the case where this extra structure is a notion of scope, for the particular case of term graphs. Scopes are abundant in programming and in logic. The guiding intuition is that the notion of scope corresponds to a notion of context-freeness. We illustrate this first by means of a string example, which although very simple already illustrates the issues involved. Consider the ad hoc context free grammar for expressions $S ::= 2 \times T$, $T ::= 3 + 1$. The obvious intended interpretation into the natural numbers of $T$ is 4 and that of $S$ is then $2 \times 4 = 8$. The standard observation is that interpretation does not commute with transformations of the grammar. In particular, substituting $3 + 1$ for $T$ in the definition of $S$ yields $S ::= 2 \times (3 + 1)$ for which one obtains $6 + 1 = 7$ as interpretation, which is not exactly the same as 8. A way to prevent such a misinterpretation is to insert parentheses first to indicate the right scope: $S ::= 2 \times (3 + 1)$.

We turn this observation around by stipulating that the notion of scope is a phenomenon that is brought about by context-free recursive specifications, for strings, terms, and graphs alike. An example for terms in programming (Lisp) is unhygienic macro expansion [5]: $(\text{or} \langle \text{exp} \rangle_1 \langle \text{exp} \rangle_2) ::= (\text{let} \langle \langle \text{exp} \rangle \rangle_1 \langle \langle \text{exp} \rangle \rangle_2)$. Expanding this macro in $(\text{or} \langle \text{nil} \rangle)$ yields $(\text{let} \langle \text{nil} \langle \text{if} \text{null} \text{if} \text{null} \rangle \rangle)$ which always yields nil due to the inadvertent capturing of $v$. In this case, a way to prevent such a misinterpretation is to insert $\lambda$s, a device from [4] for ending scopes of binders, resulting in $(\text{let} \langle \lambda v. \text{null} \langle \text{if} \text{null} \text{null} \rangle \rangle)$ avoiding that the substituted $v$ becomes bound by the $\text{let} v$ of the macro by unbinding the latter by $\lambda v$. Here we are concerned with the same phenomenon but for term graphs and their behavioral semantics. As a running example we use the following expression, which expresses a cyclic $\lambda$-term (and thereby a regular infinite $\lambda$-term) by means of the Combinatory Reduction System (CRS) inspired gletrec-notation:

\[
\begin{align*}
g\text{letrec} & \quad n() ::= \lambda x.f_1(x)f_2(x,g()) \\
f_1(x_1) & ::= \lambda x.\text{let} \alpha = x_1 \alpha \text{ in } \alpha \\
f_2(x_1,x_2) & ::= \lambda y.\text{let} \beta = x_1(x_2\beta) \text{ in } \beta \\
g() & ::= \lambda z. z \\
in & \quad n() 
\end{align*}
\]
which corresponds to the pretty printed ‘recursive graph specification’ on the left (the graph with scopes indicated by dotted lines). Our main result entails that the behavioral semantics of this specification is the same as that of the first-order term graph obtained from it, displayed on the right here. Note that in this first-order term graph additional vertices and edges between them have been inserted to delimit scopes appropriately; they play the same rôle as the parentheses in the string example and the λ in the term example. This example belongs to a particularly well-behaved subclass of the recursive graph specifications, the so-called nested term graphs, where the dependency between the nested symbols (n, f₁, f₂, g in the example) is tree-like. In particular, the first-order term graph can be interpreted as a λ-term graph [3] or a higher-order term graph [1]. However, our results pertain to specifications with arbitrary dependencies, allowing for both sharing and cyclicity, such as:

\[
\begin{align*}
g\text{letrec } f & \::= \lambda x. g(x) \\
g(X_1) & \::= \lambda x. g(y) X_1 \\
in & f()
\end{align*}
\]

**Ordinary term graphs.** Let Σ be a (first-order) signature for function symbols with arity function \(ar : \Sigma \rightarrow \mathbb{N}\). A term graph over Σ (a Σ-term-graph) is a tuple \(\langle V, \text{lab}, \text{args}, \text{root} \rangle\) where \(V\) is a set of vertices, \(\text{lab} : V \rightarrow \Sigma\) the (vertex) label function, \(\text{args} : V \rightarrow V^*\) the argument function that maps every vertex \(v\) to the word \(\text{args}(v)\) consisting of the \(ar(\text{lab}(v))\) successor vertices of \(v\) (hence it holds \(|\text{args}(v)| = ar(\text{lab}(v))\)), and \(\text{root} \in V\) is the root of the term graph. Such a term graph is called root-connected if every vertex is reachable from the root by a path that arises by repeatedly going from a vertex to one of its successors. By \(\text{TG}(\Sigma)\) we denote the class of all root-connected term graphs over Σ. Note: By a ‘term graph’ we will mean by default a root-connected term graph.

A rooted ARS is the extension of an abstract rewriting system (ARS) \(\rightarrow\) by specifying one of its objects as designated root. A rooted ARS \(\rightarrow\) with objects \(A\) and root \(a\) is called a tree if \(\rightarrow\) is acyclic (there is no \(x \in A\) such that \(x \rightarrow^* x\)), co-deterministic (for every \(x \in A\) there is at most one step of \(\rightarrow\) with target \(x\)), and root-connected (every element \(x \in A\) is reachable from \(a\) via a sequence of steps of \(\rightarrow\), i.e. \(a \rightarrow^* x\)).

**Nested Term Graphs** A signature for nested term graphs (an ntg-signature) is a signature \(\Sigma\) for term graphs that is partitioned into a part \(\Sigma_{at}\) for atomic symbols, and a part \(\Sigma_{ne}\) for nested symbols, that is, \(\Sigma = \Sigma_{at} \cup \Sigma_{ne}\) and \(\Sigma_{at} \cap \Sigma_{ne} = \emptyset\). In addition to a given signature \(\Sigma\) for nested term graphs we always assume additional interface symbols from the set \(IO = I \cup O\), where \(I = \{i\}\) consists of a single unary input symbol (symbolizing an input edge into a term graph), and \(O = \{o_1, o_2, o_3, \ldots\}\) a countably infinite set of output symbols with arity zero (symbolizing a numbered output edge from a term graph).
Definition 1 (recursive specifications for nested term graphs). Let $\Sigma$ be a signature for nested term graphs. A recursive (nested term) graph specification (an rgs) over $\Sigma$ is a tuple $\langle rec, r \rangle$, where:

- $rec : \Sigma_{ne} \to TG(\Sigma \cup IO)$ is the specification function that maps a nested function symbol $f \in \Sigma_{ne}$ with $ar(f) = k$ to a term graph $rec(f) = F \in TG(\Sigma \cup \{i_0, i_1, \ldots, i_k\})$ that has precisely one vertex labeled by $i$, the root, and that contains precisely one vertex labeled by the $o_i$ for each $i \in \{1, \ldots, k\}$;
- $r \in \Sigma_{ne}$, a nullary symbol (that is, $ar(r) = 0$), is the root symbol.

For such an rgs $\mathcal{R} = \langle rec, r \rangle$ over $\Sigma$, the rooted dependency ARS $\rightsquigarrow$ of $\mathcal{R}$ has as objects the nested symbols in $\Sigma_{ne}$, it has root $r$, and the following steps: for all $f, g \in \Sigma_{ne}$ such that $g$ occurs in the term graph $rec(f)$ at position $p$ there is a step $p : f \rightsquigarrow g$.

Example 2. Let $\Sigma_{at} = \{\lambda/1, \@/2, \nu/0\}$ for expressing $\lambda$-terms as term graphs.

(i) Let $\Sigma_{0,ne} = \{r_0/0, f_2/2, g/0\}$. Then $\mathcal{R}_0 = \langle rec_0, r_0 \rangle$, where $rec_0 : \Sigma_{0,ne} \to TG(\Sigma \cup IO)$ is defined by $r_0 \mapsto R_0, f_2 \mapsto F_2$, and $g \mapsto G$ as shown in Fig. 1 is an rgs.

(ii) Let $\Sigma_{ne} = \{n/0, f_1/1, f_2/2, g/0\}$. Then $\langle rec, n \rangle$, where $rec : \Sigma_{ne} \to TG(\Sigma \cup IO)$ is defined by $n \mapsto N, f_1 \mapsto F_1, f_2 \mapsto F_2$, and $g \mapsto G$ as shown in Fig. 1 is an rgs.

Definition 3 (nested term graphs). Let $\Sigma$ be an ntg-signature. A nested term graph (an ntg) over $\Sigma$ is an rgs $\mathcal{N} = \langle rec, r \rangle$ such that the rooted dependency ARS $\rightsquigarrow$ is a tree. By $NG(\Sigma)$ we denote the class of nested term graphs over $\Sigma$.

Example 4. We first consider the rgs $\mathcal{R}_0 = \langle rec_0, r_0 \rangle$ from Ex. 2 (i). Its rooted dependency ARS $\rightsquigarrow$ is not a tree, because there are two steps that witness $r_0 \rightsquigarrow f_2$, namely those that are induced by the two occurrences of $f_2$ in the term graph $R_0 = rec_0(r_0)$. As a consequence, $\mathcal{R}_0$ is not a nested term graph.

But for the rgs $\mathcal{N} = \langle rec, n \rangle$ from Ex. 2 (ii), we find that the rooted dependency ARS $\rightsquigarrow$ actually is a tree with root $n$. Hence $\mathcal{N}$ is a nested term graph. For a ‘pretty print’ of $\mathcal{N}$, see the left graph on page 2.

There is an easy correspondence between nested term graphs defined by the intensional definition above, and ntgs according to an extensional definition as enrichments of ordinary term graphs. An extensional description of a nested term graph (an entg) is a tuple $\langle V, lab, args, in, out, anc, root \rangle$, where $G_0 = \langle V, lab, args, root \rangle$ is a (not necessarily root-connected) term graph over $\Sigma \cup IO$, $in : V \to V$ is a partial function that maps a vertex $v$ labeled by a nested symbol to the root of the term graph nested into $v$, $out : V \to V$ is a partial function that to every output vertex $o_i$ assigns the $i$-th successor of the vertex into which the term graph containing $o_i$ is nested, and $anc : V \to V^*$ is the ancestor function that records,
Nested Bisimulation Let $R_1 = \langle \text{rec}_1, r_1 \rangle$ and $R_2 = \langle \text{rec}_2, r_2 \rangle$ be rgs's over signatures $\Sigma_1$ and $\Sigma_2$ with the same atomic symbols. A homomorphism between rgs's $R_1$ and $R_2$ (denoted by $R_1 \rightarrow R_2$) is a function $\phi : V_1 \rightarrow V_2$ between the set of vertices of the disjoint unions $G_1$ and $G_2$ of the term graphs in the image of $\text{rec}_1$ and $\text{rec}_2$, respectively; on the vertices of $G_1$ and $G_2$ labeled with atomic, or interface labels, $\phi$ behaves like an ordinary term graph homomorphism; and on the vertices labeled with nested symbol $f_1$ and $f_2$, the following ‘interface’ clause applies (for formal details, see Appendix B):

A bisimulation between $R_1$ and $R_2$ (denoted by $R_1 \Leftrightarrow R_2$) is an rgs $R$ such that $R_1 \Leftrightarrow R \Leftrightarrow R_2$.

A nested bisimulation between rgs's $R_1$ and $R_2$ compares rgs's in a finer manner by recording also the nesting behaviour of the rgs's by means of stacks of vertices. It is defined similarly (see Appendix B) between prefixed expressions $(v_1 \cdots v_k)_v$ and $(w_1 \cdots w_k)_w$ that describe visits of the vertices $v_1$ in $G_1$ and $w_1$ in $G_2$ in the context of histories of visits as recorded by the stacks $v_1 \cdots v_k_1$ and $w_1 \cdots w_k_2$ of nested vertices of $G_1$ and $G_2$. The topmost stack element always indicates the parent nested vertex, enabling a definition by local progression clauses. We denote bisimilarity by $\Leftrightarrow_{\text{ne}}$ and functional bisimilarity by $\Leftrightarrow_{\text{ne}}$.

For an example illustrating these relations on nested term graphs, see Fig. [2]. While on rgs's, $\Leftrightarrow_{\text{ne}}$ is properly contained in $\Leftrightarrow$, these relations coincide on nested term graphs (due to the tree structure of $\Leftrightarrow$). As a consequence, also $\Leftrightarrow_{\text{ne}}$ and $\Leftrightarrow$ coincide on ntgs. If $R_1 \Leftrightarrow_{\text{ne}} R_2$ holds for rgs's $R_1$ and $R_2$, then the term graphs $\mathcal{N}(R_1)$ and $\mathcal{N}(R_2)$ specified by $R_1$ and $R_2$, respectively, are isomorphic.

Implementation by first-order term graphs Nested term graphs can be implemented in a faithful, and rather natural way as first-order term graphs. By ‘faithful’ we mean that the interpretation mapping is a retraction that preserves and reflects homomorphisms, and by ‘natural’ that it can be defined inductively on
the nesting structure. The basic idea is analogous to the interpretation of $\lambda$-ho-term-graphs as first-order $\lambda$-term-graphs in [3]. For a nested term graph $N = (\text{rec}, r)$ its first-order term graph interpretation $T(N)$ is defined over $\Sigma' = \Sigma \cup I \cup \{o/2, i_r/1, o_r/1\}$ by repeatedly replacing, starting on the term graph $\text{rec}(r)$, a vertex $v$ with a nested symbol $f$ by the term graph specification $\text{rec}(f)$ of $f$, thereby directing incoming edges at $v$ to the root $v_0$ of $\text{rec}(f)$, and replacing output vertices $\delta_i$ of $\text{rec}(f)$ by the binary symbol $o$ with the first edge targeting the $i$-th successor of $v$, and the second edge being a back-link to $v_0$. Beforehand, the ntg has been pre-processed by replacing vertices with atomic constants by vertices with fresh unary labels, and with edges to a (per nested symbol) single additional output vertex. Input/output vertices at the root level get label $i/o_r$. For an example, see page 2 for the interpretation $T(N)$ of the ntg $N$ in Ex.4.

**Theorem 5** (implementation of ntgs by first-order term graphs). Let $\Sigma$ be an ntg-signature, and $\Sigma' = \Sigma \cup I \cup \{o/2, i_r/1, o_r/1\}$. There are functions $T: \text{NG}(\Sigma) \to \text{TG}(\Sigma')$ and $N: \text{TG}(\Sigma') \to \text{NG}(\Sigma)$ between the classes of ntgs over $\Sigma$ and term graphs over $\Sigma'$ such that $N \circ T = \text{id}_{\text{NG}(\Sigma)}$, (i.e. $T$ is a retraction of $N$, and $N$ is a section of $T$) that are efficiently computable, and preserve and reflect functional bisimilarity $\approx$.

As a consequence, various well-known results for term graphs can be transferred to nested term graphs. For instance, that every nested term graph $N$, has, up to isomorphism, a unique nested term graph collapse, and that the bisimulation equivalence class of $N$ (up to isomorphism) forms a complete lattice w.r.t. $\approx$.

**Further aims** We are interested in, and have started to investigate, the following further topics:

- **Context-free graph grammars.** We want to view rgs’s as context-free graph grammars in order to recognize rgs-generated nested term graphs as context-free graphs. We expect to find a close connection.

- **Monadic formulation.** We would like to obtain a categorical semantics via algebras and coalgebras: nested term graphs as monads over some signature (cf. [2]), in order to isolate the abstract essence of the nested term graph concept, and, in particular, the implementation of nested term graphs as first-order term graphs.

- **Rewrite theory for nested term graphs.** As higher-order terms have a natural interpretation as nested term graphs, it is desirable to investigate implementations of higher-order rewriting by nested term graph rewriting, and eventually, via the correspondence above, by first-order term graph rewriting.

**References**

[1] S. Blom (2001): *Term Graph Rewriting – Syntax and Semantics*. Ph.D. thesis, Vrije Universiteit Amsterdam.

[2] N. Ghani, C. Lüth & F. de Marchi (2005): *Monads of coalgebras: rational terms and term graphs*. Mathematical Structures in Computer Science 15, pp. 433–451.

[3] C. Grabmayer & J. Rochel (2013): *Term Graph Representations for Cyclic Lambda Terms*. In: Proceedings of TERMGRAPH 2013, EPTCS 110, pp. 56–73. Extending report: arXiv:1308.1034

[4] D. Hendriks & V. van Oostrom (2003): $\lambda$. In F. Baader, editor: CADE-19, LNAI 2741. Springer, pp. 136–150.

[5] E. Kohlbecker, D.P. Friedman, M. Felleisen & B. Duba (1986): *Hygienic Macro Expansion*. In: Proceedings of the 1986 ACM Conference on LISP and Functional Programming, LFP ’86, ACM, pp. 151–161.
Appendix A: Nested term graphs, extensionally

Definition 6 (nested term graphs, extensionally). Let $\Sigma$ be a signature for nested term graphs. An extensional description of a nested term graph (an entg) over $\Sigma$ is a tuple $(V, \text{lab}, \text{args}, \text{in}, \text{out}, \text{anc}, \text{root})$, where $G_0 = (V, \text{lab}, \text{args}, \text{root})$ is a (typically not root-connected) term graph over $\Sigma \cup IO$, and additionally:

- $\text{in}: V \to V$ is the step-inside partial function that assigns to every vertex labeled with a nested symbol the root of the term graph nested into $v$;
- $\text{out}: V \to V$ is the step-outside partial function that to every output vertex $o_i$ assigns the $i$-th successor of the vertex into which the term graph containing $o_i$ is nested;
- $\text{anc}: V \to V^*$ is the ancestor function that to every vertex $v$ assigns the word $\text{anc}(v) = v_1 \cdots v_n$ made up of the vertices in which $v$ is nested: $v$ is nested in $v_n$, $v_n$ is nested in $v_{n-1}$, ..., $v_2$ is nested in $v_1$.

that satisfy, more precisely, the following conditions, for all $i, k \in \mathbb{N}$, and all $w, w', v_1, \ldots, v_k \in V$:

\begin{align*}
\text{(root)}_{\text{lab,anc}} & \quad \text{lab}(\text{root}) \in \Sigma_{\text{ne}} \land \text{anc}(\text{root}) = \varepsilon \\
\text{(defined)}_{\text{in, out}} & \quad (\text{in}(w) \iff \text{lab}(w) \in \Sigma_{\text{ne}}) \land (\text{out}(w) \iff \text{lab}(w) \in O) \\
\text{(step-into)}_{\text{in}} & \quad \text{lab}(w) \in \Sigma_{\text{ne}} \implies \{ \text{lab}(\text{in}(w)) = i \in I, \text{and in}(w) \text{ is the single vertex with label i in the sub-term-graph } G_0|_{\text{in}(w)} \text{ of } G_0 \text{ at vertex } \text{in}(w) \} \\
\text{(step-out)}_{\text{out}} & \quad \text{lab}(w) \in \Sigma_{\text{ne}} \implies \{ \text{for all } j \in \{1, \ldots, \text{ar}(\text{lab}(w))\}, \text{ G}_0|_{\text{out}(w)} \text{ contains precisely one vertex } w'_j \text{ with label } o_j, \text{ and it holds: out}(w'_j) = \text{args}(w)(j); \text{ G}_0|_{\text{out}(w)} \text{ has no other vertices with labels in O.} \}
\end{align*}

Example 7. The illustration below depicts an entg that corresponds to the nested term graph $N$ in Ex. 4 with names for vertices with nested symbols (right of such vertices), and the values of the ancestor function indicated in brackets (left of the vertices):

![Nested Term Graph Illustration](image)

Proposition 8. Every nested term graph has an extensional description. Vice versa, for every extensional description $G$ of a nested term graph there is a nested term graph for which $G$ is the extensional description.
Appendix B: Bisimulation and nested bisimulation

Homomorphism and bisimulation between recursive graph specifications and nested term graphs

**Definition 9** (homomorphism, bisimulation between rgs’s). Let \( \Sigma_1 = \Sigma_{\text{at}} \cup \Sigma_{\text{ne}} \) and \( \Sigma_2 = \Sigma_{\text{at}} \cup \Sigma_{\text{ne}} \) be an ntg-signature with the same signature \( \Sigma_{\text{at}} \) for atomic symbols. Let \( \mathcal{R}_1 = \langle \text{rec}_1, r_1 \rangle \) and \( \mathcal{R}_2 = \langle \text{rec}_2, r_2 \rangle \) be rgs’s over \( \Sigma_1 \) and \( \Sigma_2 \), respectively. Let \( G_i = \langle V_i, \text{lab}_i, \text{args}_i, \text{rt}_i, \text{outs}_i \rangle \) for \( i \in \{1, 2\} \) be the enriched (not necessarily root-connected) term graphs that arise as the disjoint union of the term graphs \( \text{rec}_i(f) \) for \( f \in \Sigma_{i,\text{ne}} \) together with functions \( \text{rt}_i : \Sigma_{i,\text{ne}} \to V_i \) and \( \text{outs}_i : \Sigma_{i,\text{ne}} \to \mathcal{P}(V_i) \) that map a nested function symbol \( f \) to its root \( \text{rt}_i(f) \) in \( G_i \), and to its set \( \text{outs}_i(f) \) of output vertices in \( G_i \), respectively.

A **homomorphism** (functional bisimulation) between \( \mathcal{R}_1 \) and \( \mathcal{R}_2 \) is a function \( \phi : V_1 \to V_2 \) such that for all \( w \in V_1 \) it holds:

\[
\phi(rt_1(r_1)) = rt_2(r_2)
\]

\[
\phi_{\text{ne}}(f_1) = f_2 \quad \text{implies} \quad \phi(rt_1(f_1)) = rt_2(f_2)
\]

\[
\text{lab}_1(w) \in \Sigma_{\text{at}} \quad \text{implies} \quad \text{lab}_2(\phi(w)) = \text{lab}_1(w) \in \Sigma_{\text{at}} \land \phi^*(\text{args}_1(w)) = \text{args}_2(\phi(w))
\]

\[
\text{lab}_1(w) \in \text{Var} \quad \text{implies} \quad \text{lab}_2(\phi(w)) \in \text{Var}
\]

\[
\text{lab}_1(w) \in \Sigma_{\text{ne}} \quad \text{implies} \quad \begin{cases} 
\text{lab}_2(\phi(w)) \in \Sigma_{\text{ne}} \land \phi(rt_1(\text{lab}_1(w))) = rt_2(\text{lab}_2(\phi(w))) \\
\quad \land \forall w' \in \text{outs}(\text{lab}_1(w)). \forall i, j \in \mathbb{N}. \\
\quad \text{lab}_1(w') = o_i \land \text{lab}_2(\phi(w')) = o_j \\
\quad \implies \phi(\text{args}_1(w)(i)) = \text{args}_2(\phi(w))(j)
\end{cases}
\]

hold, where \( \phi^* \) is the homorphic extension of \( \phi \) to a function from \( V_1^* \) to \( V_2^* \). Hereby the ‘interface’ clause \( \text{(lab, args)}_{\Sigma_{\text{ne}}} \) can be illustrated as follows:

![Diagram](image)

If there is a homomorphism \( \phi \) from \( \mathcal{R}_1 \) to \( \mathcal{R}_2 \), we write \( \mathcal{R}_1 \models_{\phi} \mathcal{R}_2 \) and \( \mathcal{R}_2 \models_{\phi} \mathcal{R}_1 \), or, dropping \( \phi \) as subscript, \( \mathcal{R}_1 \models_{\phi} \mathcal{R}_2 \) and \( \mathcal{R}_2 \models_{\phi} \mathcal{R}_1 \).

A **bisimulation** between \( \mathcal{R}_1 \) and \( \mathcal{R}_2 \) is an rgs \( \mathcal{R} \) over signature \( \Sigma = \Sigma_{\text{at}} \cup \Sigma_{\text{ne}} \) with \( \Sigma_{\text{ne}} \subseteq \Sigma_{1,\text{ne}} \times \Sigma_{2,\text{ne}} \) such that \( \mathcal{R}_1 \models_{(\pi_1, \phi)} \mathcal{R} \models_{(\pi_2, \phi)} \mathcal{R}_2 \) where \( \pi_1 \) and \( \pi_2 \) are projection functions, defined, for \( i \in \{1, 2\} \), by \( \pi_i : \Sigma_{\text{ne}} \times \Sigma_{\text{ne}} \to \Sigma_{i,\text{ne}} \), \( \langle f_1, f_2 \rangle \mapsto f_i \).

In the case of nested term graphs, one can rely on a definition for rgs’s.

**Definition 10** (homomorphism, bisimulation between ntgs). Let \( \mathcal{N}_1 \) and \( \mathcal{N}_2 \) be nested term graphs over signature \( \Sigma_1 = \Sigma_{\text{at}} \cup \Sigma_{\text{ne}} \), and \( \Sigma_2 = \Sigma_{\text{at}} \cup \Sigma_{\text{ne}} \), respectively. A **homomorphism** from \( \mathcal{N}_1 \) to \( \mathcal{N}_2 \) is a homomorphism from the rgs \( \mathcal{N}_1 \) to the rgs \( \mathcal{N}_2 \). A **bisimulation** between \( \mathcal{N}_1 \) and \( \mathcal{N}_2 \) is a bisimulation between the rgs’s \( \mathcal{N}_1 \) and \( \mathcal{N}_2 \).
For entg's, the notion of homomorphism is clear: Let $\Sigma_1$ and $\Sigma_2$ be signatures with the same atomic symbols. Let $G_1$ and $G_2$ are entg's over signatures $\Sigma_1$ and $\Sigma_2$ with the same part $\Sigma_{at}$ for atomic symbols. A homomorphism from an entg $G_1$ to an entg $G_2$ (indicated by $\varphi : G_1 \rightarrow G_2$) is a function $\phi : V_1 \rightarrow V_2$ between their vertex sets that preserves the property of being root, preserves atomic, nested, and interface labels, commutes with the partial functions $in$ and $out$, commutes with the (individual) argument function on vertices with atomic labels, and preserves the ancestor function. A bisimulation between entg’s $G_1$ and $G_2$ over $\Sigma$ (indicated by $G_1 \cong G_2$) then is an entg $G$ such that $G_1 \subseteq G \subseteq G_2$.

Definition 11 (homomorphism, bisimulation between entg’s). Let $\Sigma_1 = \Sigma_{at} \cup \Sigma_{1,ne}$ and $\Sigma_2 = \Sigma_{at} \cup \Sigma_{2,ne}$ be an ntg-signatures with the same signature $\Sigma_{at}$ for atomic symbols. Furthermore, let for each of $i \in \{1,2\}$, $G_i = \langle V_i, lab_i, argsi, anc_i, in_i, out_i, root_i \rangle$ be an entg over signature $\Sigma$.

A homomorphism (functional bisimulation) between $G_1$ and $G_2$ is a $(\Sigma_{ne}, \Sigma_{at}, Var)$-respecting morphism $\phi : V_1 \rightarrow V_2$ between the structures $G_1$ and $G_2$ in the sense that for all $w \in V_1$ the conditions:

$$\phi(root_1) = root_2 \land \phi^*(anc_1(w)) = anc_2(\phi(w)) \quad (\text{root, anc})$$

$$lab_1(w) \in \Sigma_{at} \implies lab_2(\phi(w)) = lab_1(w) \in \Sigma_{at} \land \phi^*(args_1(w)) = args_2(\phi(w)) \quad (\text{lab, args})$$

$$lab_1(w) \in \Sigma_{1,ne} \implies lab_1(\phi(w)) \in \Sigma_{2,ne} \land \phi(in_1(w)) = in_2(\phi(w)) \quad (\text{lab, in})$$

$$lab_1(w) \in Var \implies lab_1(\phi(w)) \in Var \land \phi(out_1(w)) = out_2(\phi(w)) \quad (\text{lab, out})$$

hold, where $\phi^*$ is the homorphic extension of $\phi$ to a function from $V_1^+ \rightarrow V_2^+$. If there is a homomorphism $\phi$ from $G_1$ to $G_2$, we write $G_1 \cong \phi G_2$ and $G_2 \cong \phi G_1$, or, dropping $\phi$ as subscript, $G_1 \cong G_2$ and $G_2 \cong G_1$.

A bisimulation between $G_1$ and $G_2$ is an entg $B = \langle B, lab, argsi, in, out, anc, root \rangle$ where $B \subseteq V_1 \times V_2$ and root = $\langle root_1, root_2 \rangle$ such that $G_1 \cong \pi_1 B \cong \pi_2 G_2$ where $\pi_1$ and $\pi_2$ are projection functions, defined, for $i \in \{1,2\}$, by $\pi_i : V_1 \times V_2 \rightarrow V_i$, $(v_1, v_2) \mapsto v_i$. If there exists a bisimulation $B$ between $G_1$ and $G_2$, then we write $G_1 \cong B G_2$, or just $G_1 \cong G_2$.

Proposition 12. The notions of homomorphism and bisimilarity for ntgs correspond to the notions of homomorphism and bisimilarity for entg’s, via the mappings between these concepts stated in Prop. [8]

Nested bisimulation and nested homomorphism between rgs’s and ntgs

Definition 13 (nested bisimulation and nested homomorphism between rgs’s and ntgs). Let $\Sigma_1 = \Sigma_{at} \cup \Sigma_{1,ne}$ and $\Sigma_2 = \Sigma_{at} \cup \Sigma_{2,ne}$ be ntg-signatures with the same signature $\Sigma_{at}$ for atomic symbols. Let $R_1 = \langle rec_1, r_1 \rangle$ and $R_2 = \langle rec_2, r_2 \rangle$ be rgs’s over $\Sigma_1$ and $\Sigma_2$, respectively. Let $G_i = \langle V_i, lab_i, argsi, root_i, r_i \rangle$ for $i \in \{1,2\}$ be the enriched (typically not root-connected) term graph that arises as the disjoint union of the term graphs $rec_i(f)$ for $f \in \Sigma_{at}$ such that its root $root_i \in V_i$ is the root of $rec_i(r_i)$, and with as enrichment the function $r_i : \Sigma_{i,ne} \rightarrow V_i$ that maps a nested function symbol $f \in \Sigma_{i,ne}$ to its root $r_i(f)$ in $G_i$ (as a consequence it holds that $root_i = r_i(r_i)$).

A nested bisimulation between $R_1$ and $R_2$ is a relation $B_{ne} \subseteq V_1^+ \times V_2^+ \times V_2 \times V_2$, for which we will indicate elements $(v_1 \ldots v_k, v, w_1 \ldots w_k, v) \in B_{ne}$ as $(v_1 \ldots v_k)v B_{ne} (w_1 \ldots w_k)w$, with the following properties, for all $i, j, k, k_1, k_2 \in \mathbb{N}, v, v_1, \ldots, v_k \in V_1, w, w_1, \ldots, w_k \in V_2, f \in \Sigma_{at}, f_i \in \Sigma_{1,ne}$, and $f_2 \in \Sigma_{2,ne}$:

$$\begin{align*}
\text{(root)}_{\Sigma_{at}} & \quad (\text{equivalently: } (\text{root})_{\Sigma_{at}} (\text{root})_{\Sigma_{at}}) \\
\text{(lab)}_{\Sigma_{at}} & \quad (v_1 \ldots v_k)v B_{ne} (w_1 \ldots w_k)w \implies (lab(v) = lab(w) \in \Sigma_{at}) \lor (lab(v) \in \Sigma_{1,ne} \land lab(w) \in \Sigma_{2,ne}) \\
& \quad \lor (lab(v) = lab(w) \in \Sigma_{at} \land lab(w) \in \Sigma_{at}) \\
\text{(args)}_{\Sigma_{at}} & \quad (v_1 \ldots v_k)v B_{ne} (w_1 \ldots w_k)w \land lab_1(v) = lab_2(w) = f \in \Sigma_{at} \\
& \quad \implies \forall i \in \{1, \ldots, \text{ar}(f)\}. (v_1 \ldots v_k)argsi(v)(i) B_{ne} (w_1 \ldots w_k)args_2(w)(i)
\end{align*}$$
Furthermore, Theorem 16.
Let \( \Sigma \), Lemma 18. exist and to be an ntg by Lemma 15.

Over all the bisimulations between \( \Sigma \), Lemma 19.

Relationships between homomorphism/bisimilarity and nested homomorphism/nested bisimilarity

A nested homomorphism from \( R_1 \) to \( R_2 \) is a partial function \( \phi_{\text{ne}} : V_1^* \times V_1 \rightarrow V_2^* \times V_2 \) such that the relation \( \{(v_1 \cdots v_n, v_1 \cdots v_n, w) \in V_1^* \times V_1 \times V_1^* \times V_2 | \phi_{\text{ne}}((v_1 \cdots v_n, v)) \downarrow = (w_1 \cdots w_n, w)\} \) is a nested bisimulation between \( R_1 \) and \( R_2 \). If there exists a nested homomorphism between \( R_1 \) and \( R_2 \), then we write \( R_1 \bowtie_{\text{ne}} R_2 \), or just \( R_1 \bowtie_{\text{ne}} R_2 \). A nested homomorphism \( \phi_{\text{ne}} \) between \( R_1 \) and \( R_2 \) is called a nested isomorphism if \( \phi_{\text{ne}} \) is injective. If there exists a nested isomorphism between \( R_1 \) and \( R_2 \), then we write \( R_1 \bowtie_{\phi} R_2 \), or just \( R_1 \bowtie R_2 \).

Proposition 14. Let \( \Sigma_{\text{at}} \) be a signature for atomic symbols. Nested bisimilarity \( \bowtie_{\text{ne}} \) is an equivalence relation on the class of rgs’s over signatures \( \Sigma \) with atomic part \( \Sigma_{\text{at}} \). Functional nested bisimilarity \( \bowtie_{\text{ne}} \) on this class of rgs’s is reflexive and transitive.

Lemma 15. Let \( \Sigma_1 \) and \( \Sigma_2 \) ntg-signatures with the same part \( \Sigma_{\text{at}} \) for atomic symbols. Let \( R_1 \) an rgs over \( \Sigma_1 \), and \( R_2 \) an rgs over \( \Sigma_2 \) such that \( R_1 \bowtie_{\text{ne}} B_{\text{ne}} \bowtie_{\text{ne}} R_2 \) for a nested bisimulation \( B_{\text{ne}} \).

Then there exists an rgs \( R_{\bowtie_{\text{ne}}} \) over a signature \( \Sigma_{\bowtie_{\text{ne}}} \) with atomic part \( \Sigma_{\text{at}} \) such that \( R_1 \bowtie_{\text{ne}} R_{\bowtie_{\text{ne}}} \bowtie_{\text{ne}} R_2 \). Furthermore \( R_{\bowtie_{\text{ne}}} \) can be defined in such a way that it is a nested term graph in case that \( B_{\text{ne}} \) is the minimal nested bisimulation between \( R_1 \) and \( R_2 \).

Theorem 16. Let \( \Sigma_1 \) and \( \Sigma_2 \) ntg-signatures with the same part \( \Sigma_{\text{at}} \) for atomic symbols. Then for all rgs’s \( R_1 \) over \( \Sigma_1 \), and for all rgs’s \( R_2 \) over \( \Sigma_2 \) it holds:

\[
R_1 \bowtie_{\text{ne}} R_2 \iff \exists \text{ rgs's } R. \ R_1 \bowtie_{\text{ne}} R \bowtie_{\text{ne}} R_2.
\]

Definition 17 (nested term graph specified by an rgs). Let \( \Sigma \) be an ntg-signature. Furthermore, let \( R \) an rgs over \( \Sigma \), and \( B_{\text{ne}} \) the minimal nested bisimulation between \( R \) and itself (which exists as the intersection of all the bisimulations between \( R \) and itself, of which there is at least one according due to Prop. 14).

By the nested term graph \( N(R) \) that is specified by the rgs \( R \) we mean the rgs \( R_{B_{\text{ne}}} \) that is stated to exist and to be an ntg by Lemma 15.

Lemma 18. For every rgs \( R \) it holds: \( N(R) \bowtie_{\text{ne}} R \).

Relationships between homomorphism/bisimilarity and nested homomorphism/nested bisimilarity

Lemma 19. Let \( \Sigma_1 \) and \( \Sigma_2 \) be ntg-signatures with the same part \( \Sigma_{\text{at}} \) for atomic symbols. Let \( R_1 \) be an rgs over \( \Sigma_1 \), and \( R_2 \) an rgs over \( \Sigma_2 \). Then the following statements hold:

(i) \( R_1 \bowtie R_2 \implies R_1 \bowtie_{\text{ne}} R_2 \).

(ii) The converse implication of (i) does not hold in general.

(iii) \( R_1 \bowtie R_2 \implies R_1 \bowtie_{\text{ne}} R_2 \).

Lemma 20. Let \( \Sigma_1 \) and \( \Sigma_2 \) be ntg-signatures with the same part \( \Sigma_{\text{at}} \) for atomic symbols. Then for all rgs’s \( R \) over \( \Sigma_1 \), and \( S \) over \( \Sigma_2 \), and for all nested term graphs \( N \) over \( \Sigma_1 \), and \( M \) over \( \Sigma_2 \), the following implications hold:
(i) \( N \mapsto^{ne} S \Rightarrow N \mapsto S \).

(ii) \( N \leftrightarrow^{ne} M \Rightarrow N \leftrightarrow M \);

(iii) \( N \leftrightarrow M \Rightarrow N \simeq M \);

**Theorem 21.** Let \( \Sigma_1 \) and \( \Sigma_2 \) ntg-signatures with the same part \( \Sigma_{at} \) for atomic symbols. For all rgs’s \( R_1 \) over \( \Sigma_1 \), and for all rgs’s \( R_2 \) over \( \Sigma_2 \), the following statements are equivalent:

(i) \( R_1 \leftrightarrow R_2 \);

(ii) \( R_1 \leftrightarrow^{ne} R_2 \);

(iii) \( N(R_1) \simeq N(R_2) \).

Thus for rgs’s, bisimilarity coincides with nested bisimilarity, and also with the property that specified nested term graphs are isomorphic.
Appendix C: Implementation by first-order term graphs

Nested term graphs can be implemented as first-order term graphs via an easy effective procedure. Alternatively as described before Thm. 5, this procedure can also be described easily when starting on an entg representation $G = (V, \text{lab}, \text{args}, \text{in}, \text{out}, \text{anc}, \text{root})$ of a nested term graph $\mathcal{N}$ over signature $\Sigma$. Then the first-order term graph $T(\mathcal{N})$ over $\Sigma \cup \{o/\text{left}2, i/\text{left}1, o/\text{right}1\}$ is obtained from $G$ by the following steps:

1. remove every vertex $v$ with a nested symbol, redirect incoming edges at $v$ to the vertex $\text{in}(v)$;
2. relabel every output vertex $w$ with nullary label $o$ by the binary label $o$, directing the first edge (which becomes a back-link) from $w$ to $\text{out}(w)$, and the second edge from $w$ to $v_n$ where $\text{anc}(w) = v_1 \ldots v_n$, which has label $i$;
3. relabel the input vertex (with label $i$) at the root by the special unary symbol $i_r$;
4. replace every vertex with a nullary symbol by a vertex labeled with a fresh unary symbol whose outgoing edge targets a chain of fresh binary output vertices whose back-links target respective input vertices of the nesting structure; the outermost output vertex has label $o_r$ with a backlink to $i_r$.

See Example 6 for an example of this procedure on the following two pages.

We recall the following theorem.

**Theorem 5** (implementation of ntgs by first-order term graphs). Let $\Sigma$ be an ntg-signature, and $\Sigma' = \Sigma \cup \{i/\text{left}1, o/\text{left}2, i_r/\text{left}1, o_r/\text{left}1\}$. There are functions $T : \text{NG}(\Sigma) \rightarrow \text{TG}(\Sigma')$ and $N : \text{TG}(\Sigma') \rightarrow \text{NG}(\Sigma)$ between the classes of ntgs over $\Sigma$ and term graphs over $\Sigma'$ such that $N \circ T = \text{id}_{\text{NG}(\Sigma)}$, (i.e. $T$ is a retraction of $N$, and $N$ is a section of $T$) that are efficiently computable, and preserve and reflect functional bisimilarity $\rightarrow$.

As a rather direct consequence of this theorem, various well-known results for term graphs can be transferred to nested term graphs. As an example, we formulate the following corollary.

**Corollary.** Let $\mathcal{N}$ be a nested term graph. $\mathcal{N}$ has, up to isomorphism, a unique nested term graph collapse. The bisimulation equivalence class of $\mathcal{N}$ (up to isomorphism) forms a complete lattice w.r.t. $\rightarrow$.

---

1There is a subtle difference in the definition of $T(\mathcal{N})$ here in comparison with the one before Thm. 5 concerning the removal of vertices with constant symbols: here this removal happens, in step (iv), as a kind of post-processing of the interpretation; in the earlier definition it was performed as a pre-processing step on the nested term graph. However, the post-processing step used here can be modified suitably in order to obtain the same result (up to isomorphism) as by the procedure in the earlier definition.
**Example 6.** We showcase the transformation of a nested term graph into a representing first-order term graph for the example of the nested term graph $\mathcal{N}$ in Ex. 4. We start from the entg representation of $\mathcal{N}$ displayed on page 4:

By (i) removing every vertex $v$ with a nested symbol, redirecting incoming edges at $v$ to the vertex $in(v)$, and (ii) relabeling every output vertex $w$ with the nullary label $o_i$ by the binary label $o$, thereby directing the first edge from $w$ to $out(w)$ (we deal with the second edge later), we obtain:

Then by (ii)' directing the second edge from an output vertex $w$ to be a backlink from $w$ to the corresponding input vertex, that is, the vertex $v_n$ where $anc(w) = v_1 \ldots v_n$ (then $v_n$ is guaranteed to be an input vertex, i.e. it is labeled by $i$), we obtain the term graph:
(The downscaled term graph above repeats the last one from the previous page.) Finally, (iii) by relabeling
the input vertex at the root with the special symbol $i_r$, (iv) by changing the arity of variable vertices (labeled
by $v$) from zero to two, by directing the second edge back to the corresponding input vertex thereby creating
a backlink (as above the ancestor function can be used for this purpose), and by directing the first edge to
additional vertices with unary label $o_r$ that link back to the input vertex at the root, we obtain:

This term graph is a close variant of the term graph right on page 2 and here below (below two dotted
edges indicate corresponding edges in the term graph right on page 2). Note that in three cases backlinks
(once from an $o$-vertex, and twice from an $o_r$-vertex) are only indicated here. In order to facilitate a quick
structural comparison with the input of this procedure, we also show again, below left, the ‘pretty print’ of
the nested term graph $\mathcal{N}$ (as left in the figure on page 2).