Nested Term Graphs

(work in progress)

Clemens Grabmayer and Vincent van Oostrom

Computer Science (VU University Amsterdam) and Philosophy (Utrecht University)

TERMGRAPH 2014

13 July 2014
nested

‘a group of objects made to fit close together or one within another’

\[ x = \sqrt{2 + \sqrt{2 + \sqrt{2 + \sqrt{2 + \ldots}}}} \]

for \( i = 0 \) to \( 9 \) do
  for \( j = 0 \) to \( 9 \) do
    for \( k = 0 \) to \( 9 \) do
      \( \text{sum} = \text{sum} + i \times 100 + j \times 10 + k + 1; \)
nested term graphs

- motivation
  - an implementation of higher-order term graphs as first-order term graphs
  - representing nested scope structure of terms in $\lambda$ or in $\lambda_{\text{letrec}}$

- definitions
  - intensional definition as: recursive graph specifications
  - extensional definition as: enriched first-order term graphs

- bisimulation, and nested bisimulation

- implementation as first-order term graphs

- further investigations and aims
higher-order as first-order term graphs \[\text{[TERMGRAPH 2013]}\]

\[
\text{let } f = \lambda x. (\lambda y. f x) x \text{ in } f
\]

higher-order term graph [Blom '03]

higher-order term graph (abstraction-prefix funct.)

first-order term graph

CG, Jan Rochel:

\begin{itemize}
  \item \textit{Term Graph Representations for Cyclic Lambda Terms}, TG 2013.
  \item \textit{Maximal Sharing in the Lambda Calculus with Letrec}, ICFP 2014.
\end{itemize}
higher-order as first-order term graphs \[\text{[TERMGRAPH 2013]}\]

let \( f = \lambda x. (\lambda y. f x) x \) in \( f \)

higher-order term graph \[\text{[Blom '03]}\]

higher-order term graph (abstraction-prefix funct.)

first-order term graph

CG, Jan Rochel:

- *Term Graph Representations for Cyclic Lambda Terms*, TG 2013.
nested scopes in $\lambda$-terms

First-order term graph over $\Sigma = \{\lambda/1, \emptyset/2, v/0\}$
nested scopes in \( \lambda \)-terms

\[
\lambda x.(\lambda y. \text{let } \alpha = x\alpha \text{ in } \alpha)(\lambda z. \text{let } \beta = x(\lambda w. w)\beta \text{ in } \beta)
\]
nested scopes in $\lambda$-terms

$$\lambda x. (\lambda y. \text{let } \alpha = x\alpha \text{ in } \alpha)(\lambda z. \text{let } \beta = x(\lambda w. w) \beta \text{ in } \beta)$$
nested scopes in $\lambda$-terms

$$\lambda x. (\lambda y. \text{let } \alpha = x\alpha \text{ in } \alpha) (\lambda z. \text{let } \beta = x(\lambda w. w) \beta \text{ in } \beta)$$
nested scopes in $\lambda$-terms

$\lambda x. (\lambda y. \text{let } \alpha = x\alpha \text{ in } \alpha)(\lambda z. \text{let } \beta = x(\lambda w. w)\beta \text{ in } \beta)$
nested scopes in $\lambda$-terms

$$\lambda x. (\lambda y. \text{let } \alpha = x\alpha \text{ in } \alpha) (\lambda z. \text{let } \beta = x(\lambda w. w) \beta \text{ in } \beta)$$
nested scopes in $\lambda$-terms

$$\lambda x. (\lambda y. \text{let } \alpha = x\alpha \text{ in } \alpha)(\lambda z. \text{let } \beta = x(\lambda w. w)\beta \text{ in } \beta)$$
nested scopes in λ-terms
nested scopes $\rightarrow$ nested term graph
nested term graph

\[ \text{gletrec} \]

\[ n() = \lambda x. f_1(x)f_2(x, g()) \]
\[ f_1(X_1) = \lambda x. \text{let } \alpha = X_1 \alpha \text{ in } \alpha \]
\[ f_2(X_1, X_2) = \lambda y. \text{let } \beta = X_1(X_2 \beta) \text{ in } \beta \]
\[ g() = \lambda z. z \]

in

\[ n() \]
Nested Term Graphs

Grabmayer, van Oostrom
A signature for nested term graphs (ntg-signature) is a signature $\Sigma$ that is partitioned into:

- **atomic** symbol alphabet $\Sigma_{at}$
- **nested** symbol alphabet $\Sigma_{ne}$

Additionally used:

- **interface** symbols alphabet $IO = I \cup O$
  - $I = \{i\}$ with $i$ unary
  - $O = \{o_1, o_2, o_3, \ldots\}$ with $o_i$ nullary
recursive graph specification

Definition

Let $\Sigma$ be an ntg-signature.
A **recursive graph specification** (a rgs) $\mathcal{R} = \langle rec, r \rangle$ consists of:

1. **specification function**
   
   $rec : \Sigma_{ne} \rightarrow TG(\Sigma \cup IO)$
   
   $f/k \mapsto rec(f) = F \in TG(\Sigma \cup \{i, o_1, \ldots, o_k\})$

   where $F$ contains precisely one vertex labeled by $i$, the root, and one vertex each labeled by $o_i$, for $i \in \{1, \ldots, k\}$;

2. nullary **root symbol** $r \in \Sigma_{ne}$.

rooted dependency $ARS \leftarrow$ of $\mathcal{R}$:

- objects: nested symbols in $\Sigma_{ne}$
- steps: for all $f, g \in \Sigma_{ne}$:
  
  $p : f \leftarrow g \iff g$ occurs in the term graph $rec(f)$ at position $p$
recursive graph specification

\[\Sigma_{at} = \{\lambda/1, \emptyset/2, v/0\}, \Sigma_{ne} = \{r_0/0, f_2/2, g/0\}, I = \{i/1\}, O = \{o_1/0, o_2/0, \ldots\}.\]
Definition

Let $\Sigma$ be an ntg-signature. A recursive graph specification (a rgs) $R = \langle \text{rec}, r \rangle$ consists of:

- **specification function**

  $\text{rec} : \Sigma_{\text{ne}} \rightarrow TG(\Sigma \cup IO)$

  $f_k \mapsto \text{rec}(f) = F \in TG(\Sigma \cup \{i, o_1, \ldots, o_k\})$

  where $F$ contains precisely one vertex labeled by $i$, the root, and one vertex each labeled by $o_i$, for $i \in \{1, \ldots, k\}$;

- **nullary root symbol** $r \in \Sigma_{\text{ne}}$.

rooted dependency ARS $\rightarrow$ of $R$:

- objects: nested symbols in $\Sigma_{\text{ne}}$
- steps: for all $f, g \in \Sigma_{\text{ne}}$:

  $p : f \rightarrow g \iff g \text{ occurs in the term graph } \text{rec}(f) \text{ at position } p$
recursive graph specification

dependency ARS: \( f_2 \xrightarrow{r_0} g \) is a dag (but not a tree).
nested term graph: intensional definition

**Definition**

Let $\Sigma$ be an ntg-signature.

A *nested term graph* over $\Sigma$ is an rgs $\mathcal{N} = \langle \text{rec}, r \rangle$ such that the rooted dependency $\text{ARS} \rightleftharpoons$ is a tree.
nested term graph (intensionally)

dependency ARS: \( f_1 \leadsto n \leadsto f_2 \) is a tree.
nested term graph (intensionally)

dependency ARS: $f_1 \rightarrow n \leftarrow f_2 \rightarrow g$ is a tree.
nested term graph (intensionally)

infinite λ-term

(infinitely nested scopes)
nested term graph (intensionally)

infinite λ-term
(\textit{infinitely nested} scopes)

nested term graph with \textit{infinite nesting}
dependency ARS: \( f_0 \leftarrow f_1 \leftarrow f_2 \leftarrow f_3 \leftarrow \ldots \)
nested term graph (intensionally)
nested term graph: extensional definition

An extensional description of an ntg (an entg) over Σ is a term graph over Σ ∪ IO with vertex set V enriched by:

- \( in : V \rightarrow V \) (root of graph nested into \( v \))
- \( out : V \rightarrow V \) (\( i \)-th successor of vertex into which the graph containing \( v \) is nested)
- \( anc : V \rightarrow V^* \) (ancestor function): \( \text{word} \ anc(v) = v_1 v_2 \ldots v_n \) of the vertices in which \( v \) is nested.
nested term graph: extensional definition

An **extensional description** of an ntg (an *entg*) over $\Sigma$ is a term graph over $\Sigma \cup IO$ with vertex set $V$ enriched by:

- $\text{in} : V \rightarrow V$, ($v$ with nested symbol) $\mapsto$ (root of graph nested into $v$)
nested term graph: extensional definition

An extensional description of an ntg (an entg) over $\Sigma$ is a term graph over $\Sigma \cup IO$ with vertex set $V$ enriched by:

- $in : V \rightarrow V$, $(v$ with nested symbol) $\mapsto$ (root of graph nested into $v$)
- $out : V \rightarrow V$, $(v$ with output vertex $o_i$) $\mapsto$ ($i$-th successor of vertex into which the graph containing $v$ is nested)
nested term graph: extensional definition

An **extensional description** of an ntg (an entg) over $\Sigma$ is a term graph over $\Sigma \cup IO$ with vertex set $V$ enriched by:

- $in: V \rightarrow V$, ($v$ with nested symbol) $\mapsto$ (root of graph nested into $v$)
- $out: V \rightarrow V$, ($v$ with output vertex $o_i$) $\mapsto$ ($i$-th successor of vertex into which the graph containing $v$ is nested)
- $anc: V \rightarrow V^*$ **ancestor function**:
  
  $v \mapsto$ word $anc(v) = v_1 \ldots v_n$ of the vertices in which $v$ is nested
nested term graphs: intensional vs. extensional definition

Proposition

- Every nested term graph has an extensional description.
- For every entg $G$ there is a nested term graph for which $G$ is the extensional description.
bisimulation

progression condition: \( i \)-th successors of related vertices must be related

Nested Term Graphs

Grabmayer, van Oostrom
bisimulation between f-o term graphs
bisimulation between f-o term graphs

progression condition: i-th successors of related vertices must be related
bisimulation between f-o term graphs

progression condition: \( i \)-th successors of related vertices must be related
bisimulation between f-o term graphs

progression condition: \(i\)-th successors of related vertices must be related
bisimulation between f-o term graphs

progression condition: $i$-th successors of related vertices must be related
bisimulation (for intensional ntg-definition)

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be nested term graphs. Let $V_1$ the disjoint union of the vertices of term graphs in $\mathcal{N}_1$. Similar for $V_2$ w.r.t. $\mathcal{N}_2$. 
bisimulation (for intensional ntg-definition)

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be nested term graphs. Let $\mathcal{V}_1$ the disjoint union of the vertices of term graphs in $\mathcal{N}_1$. Similar for $\mathcal{V}_2$ w.r.t. $\mathcal{N}_2$.

$\mathcal{N}_1$ and $\mathcal{N}_2$ are bisimilar (denoted by $\mathcal{N}_1 \leftrightarrow \mathcal{N}_2$) if there is bisimulation between $\mathcal{N}_1$ and $\mathcal{N}_2$, i.e. a binary relation $\phi$ betw. $\mathcal{V}_1$ and $\mathcal{V}_2$ such that:

- roots are related
- related vertices either both have nested labels, or both have interface labels, or both have the same atomic label
Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be nested term graphs. Let $V_1$ the disjoint union of the vertices of term graphs in $\mathcal{N}_1$. Similar for $V_2$ w.r.t. $\mathcal{N}_2$.

$\mathcal{N}_1$ and $\mathcal{N}_2$ are **bisimilar** (denoted by $\mathcal{N}_1 \leftrightarrow \mathcal{N}_2$) if there is bisimulation between $\mathcal{N}_1$ and $\mathcal{N}_2$, i.e. a binary relation $\phi$ betw. $V_1$ and $V_2$ such that:

- roots are related
- related vertices either **both** have nested labels, or **both** have interface labels, or **both** have the same atomic label
- progression on **atomic** vertices: as for f-o term graphs
bisimulation (for intensional ntg-definition)

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be nested term graphs. Let $V_1$ the disjoint union of the vertices of term graphs in $\mathcal{N}_1$. Similar for $V_2$ w.r.t. $\mathcal{N}_2$.

$\mathcal{N}_1$ and $\mathcal{N}_2$ are bisimilar (denoted by $\mathcal{N}_1 \Leftrightarrow \mathcal{N}_2$) if there is bisimulation between $\mathcal{N}_1$ and $\mathcal{N}_2$, i.e. a binary relation $\phi$ betw. $V_1$ and $V_2$ such that:

- roots are related
- related vertices either both have nested labels, of both have interface labels, or both have the same atomic label
- progression on atomic vertices: as for f-o term graphs
- progression on nested vertices: interface clause

![Diagram of nested term graphs](image)
bisimulation (for intensional ntg-definition)

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be nested term graphs. Let $V_1$ the disjoint union of the vertices of term graphs in $\mathcal{N}_1$. Similar for $V_2$ w.r.t. $\mathcal{N}_2$.

$\mathcal{N}_1$ and $\mathcal{N}_2$ are bisimilar (denoted by $\mathcal{N}_1 \leftrightarrow \mathcal{N}_2$) if there is bisimulation between $\mathcal{N}_1$ and $\mathcal{N}_2$, i.e. a binary relation $\phi$ betw. $V_1$ and $V_2$ such that:

- roots are related
- related vertices either both have nested labels, or both have interface labels, or both have the same atomic label
- progression on atomic vertices: as for f-o term graphs
- progression on nested vertices: interface clause
bisimulation (for intensional ntg-definition)

Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be nested term graphs. Let $V_1$ the disjoint union of the vertices of term graphs in $\mathcal{N}_1$. Similar for $V_2$ w.r.t. $\mathcal{N}_2$.

$\mathcal{N}_1$ and $\mathcal{N}_2$ are **bisimilar** (denoted by $\mathcal{N}_1 \leftrightarrow \mathcal{N}_2$) if there is bisimulation between $\mathcal{N}_1$ and $\mathcal{N}_2$, i.e. a binary relation $\phi$ betw. $V_1$ and $V_2$ such that:

- roots are related
- related vertices either both have nested labels, of both have interface labels, or both have the same atomic label
- progression on **atomic** vertices: as for f-o term graphs
- progression on **nested** vertices: interface clause

![Nested Term Graphs](nested_term_graphs.png)

Grabmayer, van Oostrom
Let \( N_1 \) and \( N_2 \) be nested term graphs. Let \( V_1 \) the disjoint union of the vertices of term graphs in \( N_1 \). Similar for \( V_2 \) w.r.t. \( N_2 \).

\( N_1 \) and \( N_2 \) are **bisimilar** (denoted by \( N_1 \leftrightarrow N_2 \)) if there is bisimulation between \( N_1 \) and \( N_2 \), i.e. a binary relation \( \phi \) betw. \( V_1 \) and \( V_2 \) such that:

- roots are related
- related vertices either *both* have nested labels, or *both* have interface labels, or *both* have the same atomic label
- progression on **atomic** vertices: as for f-o term graphs
- progression on **nested** vertices: **interface clause**
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
bisimulation (for intensional ntg-definition)
nested bisimulation, and rgs’s versus ntgs

**nested bisimilarity** $\leftrightarrow^{\text{ne}}$ on rgs’s

- records nesting behaviour of rgs’s via stacks of vertices
- easy: coincides with $\leftrightarrow$ on nested term graphs
- while conceptually finer, actually coincides with $\leftrightarrow$ also on rgs’s

**nested term graph** $\mathcal{N}(\mathcal{R})$ induced by an rgs $\mathcal{R}$:

- obtained from the tree-unfolding of the dependency ARS by copying shared graph specifications

**Theorem**

Let $\Sigma_1$ and $\Sigma_2$ be ntg-signatures with same part $\Sigma_{\text{at}}$ for atomic symbols.

For all rgs’s $\mathcal{R}_1$ over $\Sigma_1$, and $\mathcal{R}_2$ over $\Sigma_2$, the following are equivalent:

(i) $\mathcal{R}_1 \leftrightarrow \mathcal{R}_2$;

(ii) $\mathcal{R}_1 \leftrightarrow^{\text{ne}} \mathcal{R}_2$;

(iii) $\mathcal{N}(\mathcal{R}_1) \sim \mathcal{N}(\mathcal{R}_2)$;
nested bisimulation, and rgs’s versus ntgs

nested bisimilarity $\leftrightarrow^{\text{ne}}$ on rgs’s

- records nesting behaviour of rgs’s via stacks of vertices
- easy: coincides with $\leftrightarrow$ on nested term graphs
- while conceptually finer, actually coincides with $\leftrightarrow$ also on rgs’s

nested term graph $\mathcal{N}(\mathcal{R})$ induced by an rgs $\mathcal{R}$:

- obtained from the tree-unfolding of the dependency ARS by copying shared graph specifications

Theorem

Let $\Sigma_1$ and $\Sigma_2$ be ntg-signatures with same part $\Sigma_{\text{at}}$ for atomic symbols.

For all rgs’s $\mathcal{R}_1$ over $\Sigma_1$, and $\mathcal{R}_2$ over $\Sigma_2$, the following are equivalent:

(i) $\mathcal{R}_1 \leftrightarrow \mathcal{R}_2$;
(ii) $\mathcal{R}_1 \leftrightarrow^{\text{ne}} \mathcal{R}_2$;
(iii) $\mathcal{N}(\mathcal{R}_1) \sim \mathcal{N}(\mathcal{R}_2)$;
nested bisimulation, and rgs’s versus ntgs

nested bisimilarity $\leftrightarrow^{\text{ne}}$ on rgs’s

- records nesting behaviour of rgs’s via stacks of vertices
- easy: coincides with $\leftrightarrow$ on nested term graphs
- while conceptually finer, actually coincides with $\leftrightarrow$ also on rgs’s

nested term graph $\mathcal{N}(\mathcal{R})$ induced by an rgs $\mathcal{R}$:

- obtained from the tree-unfolding of the dependency ARS by copying shared graph specifications

Theorem

Let $\Sigma_1$ and $\Sigma_2$ be ntg-signatures with same part $\Sigma_{\text{at}}$ for atomic symbols.

For all rgs’s $\mathcal{R}_1$ over $\Sigma_1$, and $\mathcal{R}_2$ over $\Sigma_2$, the following are equivalent:

(i) $\mathcal{R}_1 \leftrightarrow \mathcal{R}_2$;
(ii) $\mathcal{R}_1 \leftrightarrow^{\text{ne}} \mathcal{R}_2$;
(iii) $\mathcal{N}(\mathcal{R}_1) \sim \mathcal{N}(\mathcal{R}_2)$;
implementation by first-order term graphs

**Theorem**

Let $\Sigma$ be an ntg-signature, and $\Sigma' = \Sigma \cup I \cup \{o/2, i_r/1, o_r/1\}$. There is a function $T : \text{NG}(\Sigma) \to \text{TG}(\Sigma')$ such that:

1. $T$ preserves and reflects $\leftrightarrow$.
2. $T$ is efficiently computable.

Proof based on the following definition of $T$ on given nested term graph:

1. Pre-Processing: constant symbol vertices are linked to additional output vertex per nested vertex; continued outwards until top level;
2. Replacement/Adding Backlinks: starting on $\text{rec}(r)$, repeatedly replacing, a vertex $v$ with a nested symbol $f$ by the specification $\text{rec}(f)$ of $f$, thereby:
   - directing incoming edges at $v$ to the root $v_r$ of $\text{rec}(f)$
   - replacing output vertices $o_i$ of $\text{rec}(f)$ by the binary symbol $o$ with first edge to $i$-th successor of $v$, the second edge a back-link to $v_r$. 

Nested Term Graphs

Grabmayer, van Oostrom
implementation by first-order term graphs

Theorem

Let $\Sigma$ be an ntg-signature, and $\Sigma' = \Sigma \cup I \cup \{ o/2, i_r/1, o_r/1 \}$.

There is a function $T : \text{NG}(\Sigma) \to \text{TG}(\Sigma')$ such that:

(i) $T$ preserves and reflects $\leftrightarrow$.

(ii) $T$ is efficiently computable.

Proof based on the following definition of $T$ on given nested term graph:

1. Pre-Processing: constant symbol vertices are linked to additional output vertex per nested vertex; continued outwards until top level;

2. Replacement/Adding Backlinks: starting on $\text{rec}(r)$, repeatedly replacing, a vertex $v$ with a nested symbol $f$ by the specification $\text{rec}(f)$ of $f$, thereby:
   - directing incoming edges at $v$ to the root $v_r$ of $\text{rec}(f)$
   - replacing output vertices $o_i$ of $\text{rec}(f)$ by the binary symbol $o$ with first edge to $i$-th successor of $v$, the second edge a back-link to $v_r$. 

Nested Term Graphs

Grabmayer, van Oostrom
Theorem

Let $\Sigma$ be an ntg-signature, and $\Sigma' = \Sigma \cup I \cup \{ o/2, i_r/1, o_r/1 \}$.

There is a function $T : \text{NG}(\Sigma) \rightarrow \text{TG}(\Sigma')$ such that:

(i) $T$ preserves and reflects $\leftrightarrow$.

(ii) $T$ is efficiently computable.

Proof based on the following definition of $T$ on given nested term graph:

1. Pre-Processing: constant symbol vertices are linked to additional output vertex per nested vertex; continued outwards until top level;

2. Replacement/Adding Backlinks: starting on $\text{rec}(r)$, repeatedly replacing, a vertex $v$ with a nested symbol $f$ by the specification $\text{rec}(f)$ of $f$, thereby:
   - directing incoming edges at $v$ to the root $v_r$ of $\text{rec}(f)$
   - replacing output vertices $o_i$ of $\text{rec}(f)$ by the binary symbol $o$ with first edge to $i$-th successor of $v$, the second edge a back-link to $v_r$. 
implementation by first-order term graph
implementation by first-order term graph
implementation by first-order term graphs

Theorem

Let \( \Sigma \) be an ntg-signature, and \( \Sigma' = \Sigma \cup I \cup \{o/2, i_r/1, o_r/1\} \).

There is a function \( T : \text{NG}(\Sigma) \rightarrow \text{TG}(\Sigma') \) such that:

(i) \( T \) preserves and reflects \( \rightarrow \), and hence \( \leftrightarrow \).

(ii) \( T \) is efficiently computable.

Proof based on the following definition of \( T \) on given nested term graph:

1. Pre-processing: constant symbol vertices are linked to additional output vertex per nested vertex; continued outwards until top level;

2. Replacement/Adding Backlinks: starting on \( \text{rec}(r) \), repeatedly replacing, a vertex \( v \) with a nested symbol \( f \) by the specification \( \text{rec}(f) \) of \( f \), thereby:
   - directing incoming edges at \( v \) to the root \( v_r \) of \( \text{rec}(f) \)
   - replacing output vertices \( o_i \) of \( \text{rec}(f) \) by the binary symbol \( o \) with first edge to \( i \)-th successor of \( v \), the second edge a back-link to \( v_r \).
implementation by first-order term graph
implementation by first-order term graph
implementation by first-order term graph
motivation definitions bisimulation/nested bisimulation implementation further investigations/aims

transfer of results from f-o term graphs

Corollary

Let $\mathcal{N}$ be a nested term graph.

1. $\mathcal{N}$ has, up to isomorphism, a unique nested term graph collapse.

2. The bisimulation equivalence class of $\mathcal{N}$ (up to isomorphism) forms a complete lattice w.r.t. $\to$. 
implementation fails for rgs’s
implementation fails for rgs’s
implementation fails for rgs's
implementation fails for rgs’s
implementation fails for rgs’s
implementation fails for rgs’s
further investigations and aims

- relation with similar concepts
  - proof nets and proof net reduction
- context-free graph grammars
  - view rgs’s as context-free graph grammars
  - recognize rgs-generated nested term graphs as context-free graphs
- monadic formulation
  - nested term graphs as monads over some signature
  - categorically describe the implementation as first-order term graphs
- rewrite theory
  - higher-order terms interpreted as nested term graphs
  - implementation of h-o term rewriting as:
    - ‘nested term graph rewriting’
    - then realization by f-o term graph (or port graph) rewriting
  - test-case: \( \lambda \)-calculus
further investigations and aims

- relation with similar concepts
  - proof nets and proof net reduction

- context-free graph grammars
  - view rgs’s as context-free graph grammars
  - recognize rgs-generated nested term graphs as context-free graphs

- monadic formulation
  - nested term graphs as monads over some signature
  - categorically describe the implementation as first-order term graphs

- rewrite theory
  - higher-order terms interpreted as nested term graphs
  - implementation of h-o term rewriting as:
    - ‘nested term graph rewriting’
    - then realization by f-o term graph (or port graph) rewriting
  - test-case: λ-calculus
further investigations and aims

- relation with similar concepts
  - proof nets and proof net reduction
- context-free graph grammars
  - view rgs’s as context-free graph grammars
  - recognize rgs-generated nested term graphs as context-free graphs
- monadic formulation
  - nested term graphs as monads over some signature
  - categorically describe the implementation as first-order term graphs
- rewrite theory
  - higher-order terms interpreted as nested term graphs
  - implementation of h-o term rewriting as:
    - ‘nested term graph rewriting’
    - then realization by f-o term graph (or port graph) rewriting
  - test-case: \( \lambda \)-calculus
further investigations and aims

- relation with similar concepts
  - proof nets and proof net reduction
- context-free graph grammars
  - view rgs’s as context-free graph grammars
  - recognize rgs-generated nested term graphs as context-free graphs
- monadic formulation
  - nested term graphs as monads over some signature
  - categorically describe the implementation as first-order term graphs
- rewrite theory
  - higher-order terms interpreted as nested term graphs
  - implementation of h-o term rewriting as:
    - ‘nested term graph rewriting’
    - then realization by f-o term graph (or port graph) rewriting
  - test-case: λ-calculus
implementation by first-order term graph (via entg)
implementation by first-order term graph (via entg)
implementation by first-order term graph (via entg)
implementation by first-order term graph (via entg)
implementation by first-order term graph (via entg)