Introduction

- High level approach to programing: graph rewriting based on category theory.
- Much more difficult than term rewriting (which are just trees).
High level approach to programming: graph rewriting based on category theory.

Much more difficult than term rewriting (which are just trees).

Simulation of biological phenomena.

Simulation of chemical reactions.

Study of cloning:

Typically, to produce a website, one starts by copying an existing one, then one modifies it accordingly to its will.

Social Data Anonymization techniques rely on finely tuned cloning operations.
Introduction

- High level approach to programing: graph rewriting based on category theory.
- Much more difficult than term rewriting (which are just trees).
- Simulation of biological phenomenons.
- Simulation of chemical reactions.
- Study of cloning:
  - Typically to produce a web site one starts to copy an existing one, then one modifies it accordingly to its will.
  - Social Data Anonymization techniques rely on finely tuned cloning operations.
- Need of an efficient implementation of basic categorical constructs!
Plan

1. Category Theory 101
2. Graph transformation and Categories
3. AGREE and Data Anonymization
4. Conclusion
Plan

1. Category Theory 101
2. Graph transformation and Categories
3. AGREE and Data Anonymization
4. Conclusion
Early 40’s by MacLane and Eilenberg with a unifying aim: topology and algebra.

What are the fundamental structures of those two fields?
Early 40's by MacLane and Eilenberg with a unifying aim: topology and algebra.

What are the fundamental structures of those two fields?

Results much more general than thought at first.

Set theory is just a special case of category (Lawvere).

In computer science E. Moggi was able to capture ideas previously thought to be outside of reach with the monads.

In logic J.-Y. Girard and the linear logic.

etc.
A category $\mathcal{C}$ is made of

- A collection of objects: $\text{Obj}(\mathcal{C})$
- For all $x, y \in \text{Obj}(\mathcal{C})$, a set $\text{Hom}_\mathcal{C}(x, y)$
- For all $x \in \text{Obj}(\mathcal{C})$, there is $\text{id}_x \in \text{Hom}_\mathcal{C}(x, x)$
- For all $x, y, z \in \text{Obj}(\mathcal{C})$, a function
  - $\circ : \text{Hom}_\mathcal{C}(x, y) \times \text{Hom}_\mathcal{C}(y, z) \to \text{Hom}_\mathcal{C}(y, z)$
A category $\mathcal{C}$ is made of

- A collection of objects: $\text{Obj}(\mathcal{C})$
- $\forall x, y \in \text{Obj}(\mathcal{C})$ a set $\text{Hom}_{\mathcal{C}}(x, y)$
- $\forall x \in \text{Obj}(\mathcal{C})$ there is $id_x \in \text{Hom}_{\mathcal{C}}(x, x)$
- $\forall x, y, z \in \text{Obj}(\mathcal{C})$ a function
  $\circ : \text{Hom}_{\mathcal{C}}(x, y) \times \text{Hom}_{\mathcal{C}}(y, z) \to \text{Hom}_{\mathcal{C}}(y, z)$

such that

1. **Identity:** $f \circ id = id \circ f = f$
2. **Associativity:** $(h \circ g) \circ f = h \circ (g \circ f)$
Example: Category of graphs

- Objects: $G = (V, E, s, t)$ with $s, t : E \rightarrow V$
- Morphisms: $f : G \rightarrow H$ must respect source and target functions, ie:

\[
\forall e \in E. f(s(e)) = s(f(e)) \\
\forall e \in E. f(t(e)) = t(f(e))
\]
Example: Category of graphs

- **Objects:** \( G = (V, E, s, t) \) with \( s, t : E \to V \)
- **Morphisms:** \( f : G \to H \) must respect source and target functions, ie:
  
  \[
  \forall e \in E. f(s(e)) = s(f(e)) \\
  \forall e \in E. f(t(e)) = t(f(e))
  \]

- **Exemples:**

![Diagram 1](image1)

![Diagram 2](image2)
Pullback

- Let's have: \( f : X \to Z \) and \( g : Y \to Z \)

- Fiber product: \( X \times_Z Y := \{ (x, w, y) \mid f(x) = w = g(y) \} \)
Co-construction of the pullback.

Let's have: \( f : X \rightarrow Z \) and \( g : Y \rightarrow Z \)

disjoint sum with gluing: \( X +_Z Y := X + Y + Z / \sim \)

With \( \sim \) generated by \( f(z) \sim z \sim g(z) \)
Plan

1. Category Theory 101

2. Graph transformation and Categories

3. AGREE and Data Anonymization

4. Conclusion
Rule-based term rewriting is easy: replace a tree by another one.

Much more difficult with graphs (multiple incident edges).

Categorical frameworks make it clean to express graph transformations systematically.

<table>
<thead>
<tr>
<th>PB</th>
<th>PO</th>
</tr>
</thead>
<tbody>
<tr>
<td>clone</td>
<td>merge</td>
</tr>
<tr>
<td>delete</td>
<td>add</td>
</tr>
<tr>
<td>comatch</td>
<td>match</td>
</tr>
<tr>
<td>global</td>
<td>local</td>
</tr>
</tbody>
</table>
AGREE extended rule

Extension of a framework proposed by A. Corradini, D. Duval, R. Echahed, F. Prost and L. Ribeiro [ICGT15].

**Definition (AGREE rules and matches)**

- A *rule* is

  \[
  \begin{array}{ccl}
  L & \xleftarrow{l} & K \\
  & ^{r} \searrow & \searrow^{t} \\
  & \downarrow & \\
  T_{L} & \xleftarrow{l''} & T_{K}
  \end{array}
  \]

  A *match* of such a rule is composed of a mono \(L \xrightarrow{m} G\) and a typing morphism \(G \xrightarrow{m} T_{L}\) and is such that \(l' \circ t = (\overline{m} \circ m) \circ l\).
AGREE rewrite step

Definition (AGREE rewriting)

Given $\rho = (K \xrightarrow{l} L, K \xrightarrow{r} R, K \xleftrightarrow{t} T_K, T_K \xrightarrow{l'} T_L)$ and a match $L \xleftrightarrow{m} G, G \xrightarrow{m} T_L : G \Rightarrow \rho, m \ H$ is computed as follows:

1. Span $G \xleftarrow{g} D \xrightarrow{n'} T_K$ is the pullback of $G \xrightarrow{m} T(L) \xleftarrow{l'} T_K$. Since $l' \circ t = \eta_L \circ l$ there is a unique $K \xleftarrow{n} D$.

2. $R \xrightarrow{p} H \xleftarrow{h} D$ is the pushout of $D \xleftarrow{n} K \xrightarrow{r} R$. 
AGREE rewrite step

Definition (AGREE rewriting)

Given \( \rho = (K \xrightarrow{l} L, K \xrightarrow{r} R, K \xrightarrow{t} T_K, T_K \xrightarrow{l'} T_L) \) and a match \( L \xrightarrow{m} G, G \xrightarrow{\overline{m}} T_L : G \Rightarrow \rho, m H \) is computed as follows:

1. Span \( G \xleftarrow{g} D \xrightarrow{n'} T_K \) is the pullback of \( G \xrightarrow{\overline{m}} T(L) \xleftarrow{l'} T_K \). Since \( l' \circ t = \eta_L \circ l \) there is a unique \( K \xrightarrow{n} D \).

2. \( R \xrightarrow{p} H \xleftarrow{h} D \) is the pushout of \( D \xleftarrow{n} K \xrightarrow{r} R \).
Example: copy of web pages

- The structure of a web site typically as two kind of links:
  - Internal links: file hierarchy (indirect link)
  - External links: references pointing outside of the site.
Example: copy of web pages

- The structure of a web site typically has two kinds of links:
  - Internal links: file hierarchy (indirect link)
  - External links: references pointing outside of the site.
- The cloning of a web site consists in duplicating all local files and keeping external links shared between the two copies.

should be cloned as follows

![Diagram of web site structure and cloning process]
Plan

1. Category Theory 101
2. Graph transformation and Categories
3. AGREE and Data Anonymization
4. Conclusion
Big economical issue: more or less the backbone of the business models of internet giants (Google, Facebook, Yahoo etc.).

Big political issue: Open Data Policy.
Big economical issue: more or less the backbone of the business models of internet giants (Google, Facebook, Yahoo etc.).

Big political issue: Open Data Policy.

Raw problem: given a graph $G$ we would like to produce $G'$ such that

- $\text{Stat}(G) \approx \text{Stat}(G')$
- It is not possible to reidentify nodes (or edges) of $G$ from knowing $G'$ (and some extra informations...).
Big economical issue: more or less the backbone of the business models of internet giants (Google, Facebook, Yahoo etc.).

Big political issue: Open Data Policy.

Raw problem: given a graph $G$ we would like to produce $G'$ such that

- $\text{Stat}(G) \simeq \text{Stat}(G')$
- It is not possible to reidentify nodes (or edges) of $G$ from knowing $G'$ (and some extra informations...).

Naïve approach doesn’t work : Netflix [NarayanShmatikov06].

Anonymization is an active research field ... rather artistic at the time: approaches validated through experiments.
Practical case of de-anonymization: Netflix

- Striking results of Narayan and Shamtkov 2006.
- Netflix publishes a subset of its customer data: the aim is to produce useful suggestions for movies in pay per view.

<table>
<thead>
<tr>
<th>Users</th>
<th>Movies/Marks</th>
<th>Movies/marks hidden</th>
</tr>
</thead>
<tbody>
<tr>
<td>456789</td>
<td>87/4, 998/2, 687/4</td>
<td>954/2, 486/4</td>
</tr>
<tr>
<td>654953</td>
<td>45/3, 743/3, 486/4</td>
<td>687/3, 45/4</td>
</tr>
</tbody>
</table>

- Data are simply anonymized by changing the real name to a random number.
Practical case of de-anonymization: Netflix

- Striking results of Narayan and Shamtikov 2006.

- Netflix publishes a subset of its customer data: the aim is to produce useful suggestions for movies in pay per view.

<table>
<thead>
<tr>
<th>Users</th>
<th>Movies/Marks</th>
<th>Movies/marks hidden</th>
</tr>
</thead>
<tbody>
<tr>
<td>456789</td>
<td>87/4, 998/2, 687/4</td>
<td>954/2, 486/4</td>
</tr>
<tr>
<td>654953</td>
<td>45/3, 743/3, 486/4</td>
<td>687/3, 45/4</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- Data are simply anonymized by changing the real name to a random number.

- Résults : 99% of correct de-anonymization for more than 8 marks (84% if one forget about the date when the mark was set if non mainstream movies are seen).
Problem more down to the earth than non-interference:
- Partial knowledge of the graph by the opponent.
- Active attacker (embedding fake sub graphs to re-identify them).
- Object of interests vary from one data set to another.
Problem more down to the earth than non-interference:
- Partial knowledge of the graph by the opponent.
- Active attacker (embedding fake sub graphs to re-identify them).
- Object of interests vary from one data set to another.

Hence three important points to consider:
1. Background Knowledge: What does the opponent know? Model of the opponent.
2. Privacity: what is attacked?
3. Usage: How the data is going to be analyzed?

⇒ Anonymizing techniques
Two families:
- Clustering: group together edges and nodes.
- k-anonymity (and l-diversity): there should be at least k-1 other candidates with similar features.
Two families:
  - Clustering: group together edges and nodes.
  - k-anonymity (and l-diversity): there should be at least k-1 other candidates with similar features.

We focus on the k-anonymity approach: the problem amounts to create $G'$ such that $G' = G_1 \oplus G_2 \oplus ... \oplus G_k$ such that $G_i$s are isomorphic graphs.
Social Data Anonymization: Techniques

- Two families:
  - Clustering: group together edges and nodes.
  - k-anonymity (and l-diversity): there should be at least k-1 other candidates with similar features.

- We focus on the k-anonymity approach: the problem amounts to create $G'$ such that $G' = G_1 \oplus G_2 \oplus ... \oplus G_k$ such that $G_i$s are isomorphic graphs.

- It is NP-hard to find graph transformations minimizing the editing distance between a graph and a $k$-isomorphic graph.
Two families:
- Clustering: group together edges and nodes.
- k-anonymity (and l-diversity): there should be at least k-1 other candidates with similar features.

We focus on the k-anonymity approach: the problem amounts to create $G'$ such that $G' = G_1 \oplus G_2 \oplus ... \oplus G_k$ such that $G_i$'s are isomorphic graphs.

It is NP-hard to find graph transformations minimizing the editing distance between a graph and a $k$-isomorphic graph.

One solution: select $1/k$ nodes randomly, create $k$ clones, link the clones together easy to program with AGREE approach.
Using \textit{AGREE} for \textit{k}-anonymity

- Progaming with types!
- \(L\) is just a cloud of nodes, and \(K\) is made of \(k\) clones of \(L\).
- Standard \(T_L\) is:

\[
\begin{array}{c}
\circ \\
\circ \\
\star \\
\end{array}
\]

- Simplest \(T_K\) is:

\[
\begin{array}{c}
\circ^1 \\
\circ \\
\star \\
\vdots \\
\circ^{k-1} \\
\end{array}
\]
The simplest $k$-clones are not connected to each other.
The simplest $k$-clones are not connected to each others.

AGREE allows the use of the graph structure to reconnect them:
The simplest $k$-clones are not connected to each others. AGREE allows the use of the graph structure to reconnect them:

Degree problems (nodes of degree 1).
The simplest $k$-clones are not connected to each other.

AGREE allows the use of the graph structure to reconnect them:

Degree problems (nodes of degree 1).
One possibility is to type differently the edges, eg:
Plan

1. Category Theory 101
2. Graph transformation and Categories
3. AGREE and Data Anonymization
4. Conclusion
Categorical frameworks allow simple and mathematically workable definition of complex transformations.

Only basic constructs are needed: pushouts, pullbacks.

An implementation could be very generic: labeled graphs, multigraphs, etc.

Need of efficient implementations in order to cope with real examples:

- Generic implementation of (generic) basic categorical constructs.
- Statistics on large graphs.