On finding commutative polymorphisms of core triads

Michael Wernthaler

December 23, 2020

Contents

1 TODO Task 1

We need to prove the following lemma:

Lemma 1. *Let $\mathbb{T}$ be a finite tree. The following are equivalent:*

1. *$\mathbb{T}$ is a core*
2. *$End(\mathbb{T}) = \{id\}$*
3. *$AC_{\mathbb{T}}(\mathbb{T})$ terminates with $L(v) = v$ for all vertices v of $\mathbb{T}$*

1.1 Proof

- ☐ "1. $\implies$ 2." Let $\mathbb{T}$ be a core. We claim that $End(\mathbb{T}) = \{id\}$. In contradiction to our claim, let's assume there is another homomorphism $h \in End(\mathbb{T})$ with $h \neq id$. Since $\mathbb{T}$ is a core, h must be bijective. But then there has to be at least one vertex v in $\mathbb{T} = v(e_1\xi_1, ..., e_k\xi_k), e_i \in \{0, 1\}, i \in \{1, 2, ..., k\}$ such that $\xi_a \to \xi_b$ for at least one pair $\{e_a\xi_a, e_b\xi_b\}$ with $e_a = e_b$. **TODO: But why must there be such a vertex?** But this implies that a non-bijective endomorphism of $\mathbb{T}$ must exist and $\mathbb{T}$ can't be a core. **TODO: construct it!** We see that $End(\mathbb{T})$ can not contain such a h but only id.

- ☐ "2. $\implies$ 3." **TODO** ... we can assume that AC solves $CSP(\mathbb{T})$. So if $AC_{\mathbb{T}}(\mathbb{T})$ derives $L(v)$ so that it contains another vertex $u \neq v$, there must be a homomorphism $h : \mathbb{T} \to \mathbb{T}$ such that $h(v) = u$. **TODO Why must there be such a homomorphism?** However, we know that $End(\mathbb{T}) = \{id\}$, so $L(v)$ can not contain such a vertex u, but only v. Hence $L(v) = \{v\}$.

- ☒ "3. $\implies$ 1." If $AC_{\mathbb{T}}(\mathbb{T})$ terminates with $L(v) = v$ for all vertices v of $\mathbb{T}$, we know that, if there was a homomorphism $h : \mathbb{T} \to \mathbb{T}$, h would map each vertex v to itself. We see that h is clearly an automorphism, hence $\mathbb{T}$ must be a core.

1.2 Notes

- Undirected trees are always homomorphically equivalent to a path of length 1
- Proposed by Florian:
  1. There must be a leaf u on which f is not the identity.
  2. the (unique shortest) path from u to $f(u)$ is mapped to the (unique shortest) path from $f(u)$ to $f(f(u))$
  3. (simple case) if $f(f(u)) = u$ then there is a vertex v on this path such that $f(v) = v$
  4. (in general) take the orbit of u and the paths in between, this gives a subtree, TODO show existence of v with $f(v)=v$in this subtree
  5. cut at v into pieces
  6. on the components containing u take f on other components take the identity, this gives a non-injective endomorphism

2 DONE Task 2: Arc-Consistency procedure

Implement the arc-consistency procedure such that your algorithm runs in linear time in the size of the input.

Algorithm 1: $AC_{\mathbb{T}}$ ($\mathbb{T}$ is a triad)
1 Input: digraph $\mathbb{G}$, initial lists $L : G \mapsto P(T)$ Output: Is there a homomorphism $h : \mathbb{G} \mapsto \mathbb{T}$ such that $h(v) \in L(v)$ for all $v \in G$

Algorithm 2: Algorithm for finding core triads

Input: An unsigned integer m

Output: A list of all core triads whose arms each have a length $\leq m$

```
// Finding a list of RCPs
```

pathlist $\longleftarrow$ [ ];

foreach *path* p *with length*$(p) \leq m$ **do**

  if $ACR_p(p)$ *didn't derive* $L(v) \neq v$ *for any vertex* v **then**

    put p in pathlist

```
// Assembling the RCPs to core triads
```

triadlist $\longleftarrow$ [ ];

foreach $\{p_1, p_2\}$ *in pathlist* **do**

  if $ACR_{p_1 p_2}(p_1 p_2)$ *derived* $L(v) \neq v$ *for some vertex* v **then**

    drop the pair and remember the two indices;

  else

    put $(p_1 p_2 p_3)$ in triadlist if it does not contain an index pair remembered from previous iterations;

foreach *triad* $\mathbb{T}$ *in triadlist* **do**

  if $AC_{\mathbb{T}}(\mathbb{T})$ *derived* $L(v) \neq v$ *for some vertex* v **then**

    remove $\mathbb{T}$ from triadlist;

return *triadlist*

2.1 Notes

- Can we optimize AC for paths?
- Done by implementing AC-3 for graphs

3 TODO Task 3

Write an algorithm that enumerates all core triads up to a fixed path-length.

3.1 Pseudo-Algorithm

The pseudo-code of the entire core triad generation is displayed in algorithm 2

3.2 Notes

3.2.1 Algorithm

- ACR names a "pre-coloured" Arc-Consistency Procedure that fixes the end vertex e that normally has degree 3 with $L(e) = e$
- Running AC on a path, doesn't gain helpful information about the triad

$AC \to id$	no statement possible
$ACF \to id$	no statement possible
$AC \nrightarrow id$	no statement possible
$ACF \nrightarrow id$	triad **cannot** be a core triad

- We need indexing for RCFs, to exclude dropped pairs
- "100" is an example for a path, where AC doesn't derive only id, however ("100","11","00") is still a core
- Only if ACR does not derive only id, we can drop the path

3.2.2 Misc

- We need a section in the beginning to explain notation in context of triads, e.g. $(p_1 p_2 p_3)$
- A rooted core path (RCP) p is a path for which $ACR_p(p)$ did derive $L(v) = v$ for every vertex v

3.2.3 Observations

- If $maxlength(p) = n$ then number of possible paths is $\sum_{i=1}^{n} 2^i$
- Let $\theta = (p_1 p_2 p_3)$ be a core triad, then there's no $\{p_a, p_b\}$ such that $p_a \to p_b$
- A triad with two identical arms is obviously not a core triad
- A triad with an arm that is not a RCP can't be a core triad

4 TODO Task 4

Write an algorithm that enumerates all core triads that do not have a commutative polymorphism up to a fixed path-length. For every triad $\mathbb{T}$ there is a unique homomorphism $level : \mathbb{T} \to (\mathbb{Z}, \{(n, n+1) | n \in \mathbb{Z}\})$

4.1 Notes

5 Deprecated

5.1 Task 1

- ⊟ "1. $\implies$ 2." $\mathbb{T}$ is a core. Let's assume that id is not the only endomorphism of $\mathbb{T}$, and there's an endomorphism $h \in End(\mathbb{T})$, $h \neq id$. Since $\mathbb{T}$ is a core, h must be bijective. Because there is only one path between two nodes h is induced by permutations of leaf nodes. Each group of permutated leafs induces a minimal subtree of $\mathbb{T}$ with exactly those leafs. To show: only possible permutation is id.
- ☒ "3. $\implies$ 2." It's obvious, that always $\{id\} \subseteq End(\mathbb{T})$. Since $AC_{\mathbb{T}}(\mathbb{T})$ derived $L(v) = v$ for all vertices v of $\mathbb{T}$ we know there can't be another homomorphism h for which $h(v) \neq v$, hence $End(\mathbb{T}) = \{id\}$.
- ☒ "2. $\implies$ 1." If $End(\mathbb{T}) = id$ then the only homomorphism $h : \mathbb{T} \to \mathbb{T}$ is id. id is an automorphism, hence $\mathbb{T}$ must be a core.

6 Questions

- ☐ Should the triads be saved in a separate file?
  - Visualization
  - Running a different polymorphism later

7 Todo

7.1 TODO Think of a german title

Über das finden von Siggers Polymorphismen für Core Triads

7.2 DONE Generalize AC

It should be possible to pass a unary constraint for each vertex ("pre-colouring")