

Binary Commutative Polymorphisms of Core Triads

Michael Wernthaler

January 3, 2021

Contents

1	TODO Task 1	2
1.1	Proof	2
1.2	Notes	3
2	DONE Task 2: Arc-Consistency procedure	4
2.1	Notes	4
2.2	Benchmarks	4
3	DONE Task 3	5
3.1	Algorithm	5
3.2	Notes	5
4	TODO Task 4	6
4.1	Notes	7
5	Notes	7
5.1	Deprecated	7
5.2	Program	7
5.3	Todo	7

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. Let’s assume there is another homomorphism $f \in End(\mathbb{T})$ with $f \neq id$. Knowing that \mathbb{T} is a tree we conclude there must be a leaf u on which f is not the identity. We consider $p = u_0u_1\dots u_l$ to be the unique path from u to $f(u)$, which maps to the unique path from $f(u)$ to $f(f(u))$. Our claim is that there has to be a vertex v on p for which $f(v) = v$. To show this we take the orbit of u and the paths in between.

In the simple case we suppose that $f(f(u)) = u$. This implies $f(u_i) = u_{l-i}$ for $i \in \{0, 1, \dots, l\}$. Since no double-edges are allowed, we conclude that $l = 2m$, which gives us $f(u_m) = u_m$.

For the general case, we consider the orbit of u to be of size $n \geq 3$. Because of $f(u_0) = u_l$ there is a greatest $m \leq l$ such that $f(u_i) = u_{l-i}$, for every $i \in \{0, 1, \dots, m\}$ from which follows that there must be a cyclic path from u_m to $f^n(u_m) = u_m$ of length $n(l - 2m)$. Since \mathbb{T} is a tree, we require that $n(l - 2m) = 0$. The latter equation can only be satisfied for $l = 2m$, and again we get $f(u_m) = u_m$.

Now let $\mathbb{T} = v(\xi_1, \dots, \xi_k)$ with $\xi_a \rightarrow \xi_b$ for at least one pair ξ_a, ξ_b , where ξ_a contains u and ξ_b contains $f(u)$. We then construct a nonbijective endomorphism h of \mathbb{T} by taking f on ξ_a . For every other component we define h as id . But then \mathbb{T} can’t be a core, which means our assumption was wrong and $End(\mathbb{T})$ cannot contain such a f , but only id .

⊠ “2. \implies 1.” If $End(\mathbb{T}) = id$, then the only homomorphism $h : \mathbb{T} \rightarrow \mathbb{T}$ is id , which is an automorphism. Hence \mathbb{T} must be a core.

□ “2. \implies 3.” **TODO** ...we can assume that AC solves $CSP(\mathbb{T})$. So if $AC_{\mathbb{T}}(\mathbb{T})$ derives $L(v)$ such that it contains another vertex $u \neq v$,

there must be a homomorphism $h : \mathbb{T} \rightarrow \mathbb{T}$ 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 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 $End(\mathbb{T}) = \{id\}$.

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 $f(u)$ maps 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 \mathbb{T} at v into pieces
 6. on the components containing u take f on other components take the identity, this gives a noninjective 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 $v \in G$

2.1 Notes

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

2.2 Benchmarks

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 RCAs
armlist  $\leftarrow$  [];
foreach arm  $p$  with  $\text{length}(p) \leq m$  do
    if  $ACR_p(p)$  didn't derive  $L(v) \neq v$  for any vertex  $v$  then
         $\perp$  put  $p$  in armlist

// Assembling the RCAs to core triads
triadlist  $\leftarrow$  [];
foreach  $\{p_1, p_2\}$  in armlist do
    if  $ACR_{p_1 p_2}(p_1 p_2)$  derived  $L(v) \neq v$  for some vertex  $v$  then
         $\perp$  Drop the pair and cache the two indices;

foreach triad  $\mathbb{T} = \{p_1, p_2, p_3\}$  do
    if  $\mathbb{T}$  contains a cached index pair then
         $\perp$  Drop  $\mathbb{T}$  and continue;
    if  $AC_{\mathbb{T}}(\mathbb{T})$  didn't derive  $L(v) \neq v$  for some vertex  $v$  then
         $\perp$  Put  $\mathbb{T}$  in triadlist;

return triadlist

```

3 DONE Task 3

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

3.1 Algorithm

Algorithm 2 displays the pseudo-code of the entire core triad generation.

3.2 Notes

3.2.1 Observations

- If $\text{maxlength}(p) = n$ 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 $p_a \rightarrow p_b$
- A triad with two identical arms is obviously not a core triad

- ACR names a “pre-colored” Arc-Consistency Procedure that precolors the root r that has degree 3 with $L(r) = \{r\}$
- “Rooted core arm” (RCA) names an arm a for which $ACR_a(a)$ did derive $L(v) = \{v\}$ for every vertex v
- A triad with an arm that is not a RCA can’t be a core triad

3.2.2 Algorithm

- Running AC on an arm, doesn’t gain helpful information about the triad

$AC \rightarrow id$	no statement
$AC \nrightarrow id$	no statement
$ACR \rightarrow id$	no statement
$ACR \nrightarrow id$	triad cannot be a core

- “100” serves as an example for an arm, where AC doesn’t derive only id , yet (“100”, “11”, “00”) is still a core
- Only if ACR does not derive only id , we can drop the arm

3.2.3 Optimizations

- Derive sister triads, e.g. (“01”, “0”, “11”) from (“10”, “1”, “00”)
- Optimize arc consistency for endomorphisms (don’t check for empty lists)

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

4.1 Notes

5 Notes

5.1 Deprecated

5.1.1 Task 1

- ☒ “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 $h : \mathbb{T} \rightarrow \mathbb{T}$, h would map each vertex v to itself. We see that h is obviously an automorphism, hence \mathbb{T} must be a core.

5.2 Program

5.2.1 Flags

- **-v** be verbose
- **-l** *NUM* max arm length of triad
- **-p** *NAME* polymorphism to check
- **(-s** save triads to file)

5.3 Todo

- ☐ Explain notation in context of triads, e.g. $(p_1p_2p_3)$
- ☐ Serialize triads