A niagara program with contexts can be seen as a graph, where each node is a pool, and edges between pools are operations. In this higher level representation, pools have context, which indicates where the money is coming from. Context can be seen as types, and asserting their coherence on the graph is akin to a typing problem.

This document proposes a formalization of contexts, define a typing systems, and proposes inference rules for the context of pools which are proven to give a correctly typed graph.

Mathematical preliminaries

Definitions

Let X be a set. We define a partial equivalence relation (P.E.R.) r on X as a binary relation that is symmetric and transitive. Partial reflexivity is not an axiom but a theorem stating that

$$\forall x, y, x \sim y \implies (x \sim x)$$

which stems from the fact that if there are such an x and y, by symmetry $y \sim x$ and by transitivity $x \sim y \sim x$.

Like equivalence relation, P.E.R.s correspond directly to partial partitions of X. A partial partition of X is a set of mutually disjoint subsets of X. Going from a P.E.R. r_a to a partial partition C_a and back is done by considering the equivalence classes of the relation.

To a P.E.R. r_a, we can associate its perimeter p_a, that is the set on which the equivalence relation is total (the union of the equivalence classes).

Operations on P.E.R.

Let r_a and r_b be two P.E.R.

Conjunction

The conjunction $r_c = r_a \wedge r_b$ is defined by: $x \sim_c y$ iff any of the following is true:

- $x, y \in (p_a \cap p_b)$ and $x \sim_a y$ and $x \sim_b y$
- $x, y \in p_a \setminus p_b$ and $x \sim_a y$
- $x, y \in p_b \setminus p_a$ and $x \sim_b y$

The equivalence classes C_c of r_c are given by:

$$C_c = \{c_a \cap c_b, c_a \in C_a, c_b \in C_b\} \cup \{c_a \cap \overline{p_b}, c_a \in C_a\} \cup \{c_b \cap \overline{p_a}, c_b \in C_b\}$$

Conjunction is both associative and commutative.

Disjunction

The disjunction $r_c = r_a \vee r_b$ is defined by the transitive cloture of ($x \sim y$ iif $x \sim_a y$ or $x \sim_b y$).

Projection

Let Y be a subset of X. The projection $r_c = r_a \downarrow Y$ of r_a on Y is defined by $x \sim_c y$ iff $x, y \in Y$ and $x \sim_a y$

The projection of the equivalence classes is $\{c \cap Y, c \in C_a\}$

Projection without loss

A projection is without loss iff, for all $x, y \in X$ such that $x \sim_a y$, $x \in Y$ implies $y \in Y$.

In terms of equivalence classes: for all c in C_a, $c \subseteq Y$ or $c \subseteq \overline{Y}$.

Order

We say that $r_a \leq r_b$ (is partially finer) iff for all $x, y \in X, x \sim_a y \Rightarrow x \sim_b y$.

In terms of equivalence classes, $\forall c_a \in C_a, \exists c_b \in C_b, c_a \subseteq c_b$

Remark: disjunction is an upper bound for this order relation but conjunction is ***not*** *a lower bound.*

Properties

- $(r \downarrow P) \downarrow P = r \downarrow P$
- $r \downarrow P \leq r$
- $p_{r \downarrow P} \subseteq P$
- $(r_a \wedge r_b) \downarrow P = (r_a \downarrow P) \wedge (r_b \downarrow P)$
- Let c be a class of $(\bigwedge r_i)$, then for all i, either $c \subseteq \overline{p_{r_i}}$ or there exist a class d of r_i such that $c \subseteq d$.

Type system

An operation has both an input and an input. It also has an associated projection on a set P. Our types are partial equivalence relations. The input and output have types r_i and r_o. We say that our operation is well typed if and only if:

- $r_i \downarrow P$ is without loss, and
- $r_i \downarrow P \leq r_o$

Our graph is well typed iff all its operations are well typed and for all variables v, with type r_v, and corresponding children operations C:

$$p_v \subseteq \bigcup_{c \in C} (p_{r_c} \cap P_c)$$

Because each operation is well typed, this is equivalent to

$$p_v \subseteq \bigcup_{c \in C} P_c$$

Moreover it is tightly typed if it is correctly typed and for all variables v, with type r_v, and corresponding parent operations P:

$$p_v \subseteq \bigcup_{p \in P} (p_{r_p} \cap P_p)$$

Inference

Backward inference

In backward inference we are gonna assign upper bound to each variables type with the guarantee that all correct typings are less than this upper bound and the bound itself is a correct typing.

The proof is done by induction.

The basis case is a graph with only entries that are also outputs (ie, a graph without any operation). The upper bounds for all variable is $\top$, the equivalence relation with only one class, X itself.

Let's consider a graph with bounds for each variable respecting the property above. We add "upstream" nodes to this graph playing the role of "inputs" and pouring in any of the node of the original graph. For each added node i, we consider the set of nodes it is pouring into O_i. We define:

$$r_i = \left(\bigwedge_{o \in O_i} r_o \downarrow P_o \right) \Big\downarrow \left(\bigcap_{o \in O_i} p_{r_o} \cup \overline{P_o} \right)$$

Let's prove that this makes all added operations correctly typed.

Let's consider r_i and one of its outputs in particular ω.

We must show that $r_i \downarrow P_\omega$ is without loss and less than r_ω

Without loss

Let a be a class of r_i. There exist a class b of $\left(\bigwedge_{o\in O_i} r_o \downarrow P_o\right)$ so that $a = b \cap \left(\bigcap_{o\in O_i} p_{r_o} \cup \overline{P_o}\right)$. The perimeter of $(r_\omega \downarrow P_\omega)$ is $(p_{r_\omega} \cap P_\omega)$. By one of the properties of section 1:

- either $b \subseteq \overline{p_{r_\omega} \cap P_\omega} = \overline{p_{r_\omega}} \cup \overline{P_\omega}$, in which case $a \subseteq (\overline{p_{r_\omega}} \cup \overline{P_\omega}) \cap (p_{r_\omega} \cup \overline{P_\omega}) = \overline{P_\omega}$
- or there exist a class c of $(r_\omega \downarrow P_\omega)$ so that $b \subseteq c$, in which case $a \subseteq b \subseteq c \subseteq P_\omega$.

So either $a \subseteq P_\omega$ or $a \subseteq \overline{P_\omega}$. This being true for any a, the projection is without loss. QED.

$r_i \downarrow P_\omega \leq r_\omega$

Let d be a class of $r_i \downarrow P_\omega$. There exist a class a of r_i so that $d = a \cap P_\omega$.

Taking the same a and b as above:

- either $b \subseteq \overline{p_{r_\omega} \cap P_\omega}$, in which case as above $a \subseteq \overline{P_\omega}$, and a is not part of $r_i \downarrow P_\omega$ (here actually d is the empty set, which is ill defined, we should rework all definitions to exclude the empty set but that does not change much).
- or there exist the same c as above and a is a subset of a class of $(r_\omega \downarrow P_\omega) \leq r_\omega$, so d is a subset of a class of r_ω.

QED.

Let's show the correctness of the graph as a whole.

We must show that $p_{r_i} \subseteq \bigcup_{o\in O_i}(p_{r_o} \cap P_o)$, which is immediate from the definition of p_{r_i}

$$p_{r_i} = \left(\bigcup_{o\in O_i} p_{r_o} \cap P_o\right) \bigcap \left(\bigcap_{o\in O_i} p_{r_o} \cup \overline{P_o}\right)$$

We have shown that our bounds are a correct typing of the graph as a whole. Let's show that they are indeed bounds.

From now on let's rename the bound b_i/b_o and consider a correct typing r_i/r_o of the graph as a whole. We are still doing our induction so $r_o \leq b_o$.

Let c_i be a class of r_i. Because the relations are well typed, we have that for each o in O_i: either $c \subseteq \overline{P_o}$ or there exist a class c_o of r_o so that $c_i \subseteq c_o$. By induction, there exist a class c_{b_o} of b_o so that $c_o \subseteq c_{b_o}$ and hence $c_i \subseteq c_{b_o}$.

Let's define x_o to be:

- if $c \subseteq \overline{P_o}$, $x_o = \overline{p_{b_o \downarrow P_o}} = \overline{P_o} \cup \overline{p_{b_o}}$
- if $c \subseteq P_o$, $x_o = c_{b_o} \cap P_o$

In both cases, $c \subseteq x_o$. Moreover $(\bigcap_{o \in O_i} x_o)$ is a class of $\bigwedge_{o \in O_i}(b_o \downarrow P_o)$ and because for all o, $x_o \subset (p_{b_o} \cup \overline{P_o})$, we have that $(\bigcap_{o \in O_i} x_o)$ is also a class of $b_i = \left(\bigwedge_{o \in O_i} b_o \downarrow P_o\right) \downarrow \left(\bigcap_{o \in O_i} p_{b_o} \cup \overline{P_o}\right)$. Hence c is a subset of a class of b_i.

This being true for all c, we have that $r_i \leq b_i$. QED.

Hence we have shown that our upper bound is indeed an upper bound of all correct typing of the graph.

Finally, we have the property that for any correct typing of the graph, one can change the entries type for smaller ones and still have a correct typing. Meaning that for any user provided typing of the entries smaller than the bounds, we can correctly type the graph a a whole using the bounds for the other nodes.

Tightening

Let's consider a correctly typed graph.

Let x be the first variable (in topological order) which is not tight, that is for which $r_x \nsubseteq \cup_{i \in I}(r_i \downarrow P_i)$. I is the set of input variable to x and O the set of outputs.

We define

$$r = \bigvee_{i \in I}(r_i \downarrow P_i)$$

Let's show that by setting x's type to r the graph is still correctly typed and x is now tight.

For an input i of x:

$r_i \downarrow P_i$ is still without loss and $r_i \downarrow P_i \leq r$ by definition of r, so all operations from i to x remain well typed if x type is r.

Let $\sim$ be the non transitive relation defined by $a \sim b$ iff $\exists i, x \sim_{r_i} y, x \in P_i, y \in P_i$. Let be such a, b so that $a \sim b$, and i as above. Then because $r_i \downarrow P_i \leq r_x$, we have that $a \sim_{r_x} b$. Hence, because r_x is transitive, and r is the transitive cloture of $\sim$ we have that, $r \leq r_x$. Hence, for all $o \in O$, the operation going from x to o is still well typed if x as type r.

We have $p_r \subseteq p_{r_x} \subseteq \bigcup_{o \in O} P_o$. Because all operations are well typed this is enough to show that the graph remains well typed.

Finally $p_r = \bigcup_{i \in I}(P_i \cap p_{r_i})$ so our node x is now tight.

By iterating this algorithm in topological order we can make every node tightly typed. (Because we do this in topological order we won't "detight" nodes that have already been processed.)