20230414 - Intro to SGL

SGL stands for Semantic Graph Language.

To me graphs have always had a closer connection with the representation of abstract concepts in my mind than writing them linearly. They are a better object of semantic maps (maps from syntax to 'meaning') than anything else I've seen. This makes them of great epistemological interest, in addition to their computational significance. They are also a rather neat finitist foundation of mathematics.

Semantic graphs are multi-sets of any triple of nodes, along with a subset of nodes designated pegs.

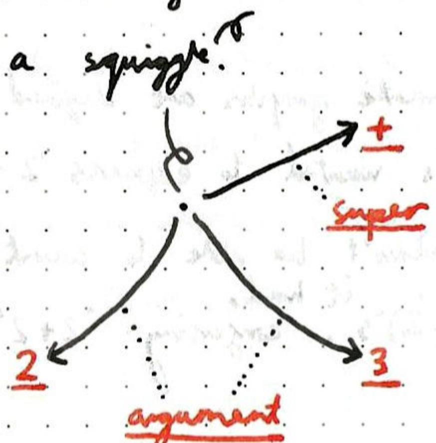
We call the triples links. We say the link (a, b, c) is outgoing from a and incoming to b & c.

Nodes have no internal beyond their outgoing links. They are each either pegs or forks.

Pegs have no outgoing links. There can be considered to have countably infinite pegs, although you should only use a finite number.

Forks on the other hand may have outgoing links. They can be considered to be defined entirely by their outgoing links.

Thats just about enough of a definition to let us look at some examples. To get us started, we will use an explicit notation. All pegs will be underlined and in orange. The remaining nodes can be considered forks. The root node will be marked with a squiggle.



This may be easier to see if we add some notational abbreviations:

Write ⟶ for: ⟶ super. And ⟶ for: ⟶ argument:

These two graphs are identical except in presentation. It involves 5 pegs (2, 3, +, argument, super), 1 fork, and 3 links. If we were to label the fork 'f' then we could write out the triples explicitly:

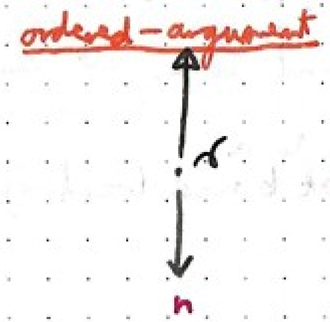
{(f, super, +),
(f, argument, 2),
(f, argument, 3)}

Now the reason why semantic graphs are defined as multisets is: suppose we wanted to express '2 + 2'. Then a regular set wouldn't be able to count how many (f, argument, 2)'s it had, confusing '2+2' with '+2'.

Noting the resemblance to syntax trees, you may be wondering if that is what, in essence, SGL is. However, seeing its simplicity, its breadth of representation, and as we'll come to next, its

variable binding - will completely change your mind.

The λ-calculus did a great job of distilling the concept of a computable function down to its bare essentials. I've found it a great source of inspiration and challenge for SGL. First some notation: When something requires ordered arguments, it is common to use a compound role. Where 'role' refers to anything used in the 2nd position of a triple. 'Compound' means that it's a fork rather than a peg. Specifically



where n is ~~0~~ a numeral. A numeral is either 0 or the successor of a numeral. Successor of m is defined analogously.

'σ(n)' may be used as short-hand.

Since they're used so frequently, I developed an even-shorter hand:

a —⊖ b means a $\xrightarrow{d(0)}$ b

a —+ b means a $\xrightarrow{d(s(0))}$ b

⋮

a —2 b

a —3 b

a —4 b

a —5 b

⋮

Additionally we will need 2 other kinds of links:

a —o b   a has local ~~owns~~ b

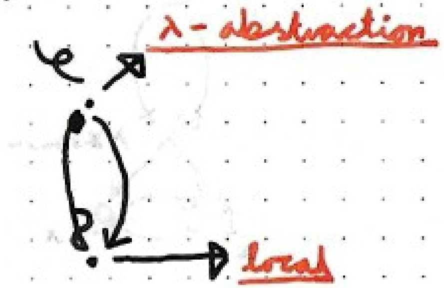
b —→ a   b has backlink to a

(note the order of a and b on the last one)

Typically, backlinks are written on the back of ~~backlo~~ ~~owning~~ local links. In that case, we can say:

a •—o b   b is bound to a
a is a ~~local~~ bound local of b

With that notation out the way, let's see some λ-expressions! The simplest closed term is identity:



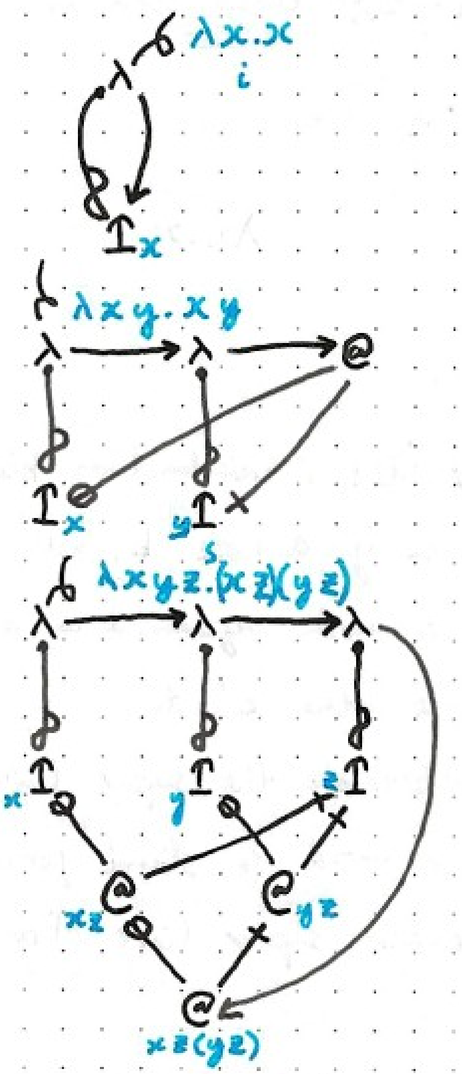
$\lambda x . x$

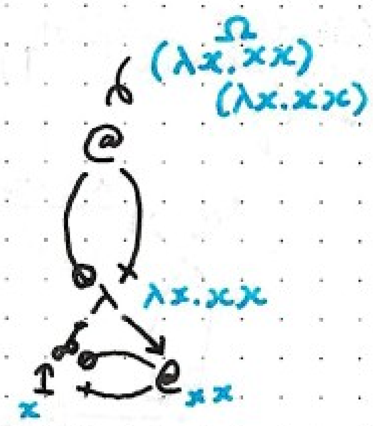
Here we can say that 'the λ-abstraction binds the local.' This is because if a node b, with b —▷ c, is the only such node referenced in a diagram then we can use 'the c' to reference it. Next we abbreviate the super links by picking a symbol or word to stand for a node with that particular super link. For λ-calculus, we need:

local	→	I
λ-abstraction	→	λ
λ-application	→	@
βλ-reduction	→	$\rightarrow_\beta$

Note that these symbol pairings are expected to be highly variable based on context and author preference. But it should always be made clear.

Thus we can write:





The blue annotations are written in the more conventional language of λ-calculus, so are written outside SGL, not part of the graph itself.
In particular, all the bound variables are just

written as locals (I), without any labels. This means <u>α-conversion is free in SGL</u>! α-conversion refers to a rule in λ-calculus stating:

Two terms are α-convertible iff one can be converted to the other purely by renaming bound variables to 'fresh' ones. Written as '$\equiv_\alpha$' or just '$\equiv$'

In SGL, this is just identity which is just isomorphism.

A node a can reach the node b iff

$$\exists (c_i),(d_i) : L(a, c_0, d_0) \wedge (b \equiv c_n \cdot \vee\ b \equiv d_n \cdot)$$
$$\wedge\ \forall i \in [0..n) : L(c_i, c_{i^+}, d_{i^+}) \vee L(d_i, c_{i^+}, d_{i^+})$$

That is a can reach b iff there is an <u>outward path</u> from a to b. Usually just called a path.
Call the <u>reach set</u> the multiset of links reachable from a node. So a can reach b and $L(b, c, d) \times n$ these n links are also in the reach set, and vice versa.

$$\forall a,b,c,d,n : Reach(a,b) \Rightarrow (L!(b,c,d,n) \Leftrightarrow RL!(a,b,c,d,n))$$

Then we define SGL identity '$\equiv$' formally as a pair of bij-ections:

Two nodes a, b are identical iff there is a permutation φ between nodes in SGL-space such that all the following hold:

(i) φ is a bijection

(ii) $\varphi(a) \equiv b$

(iii) $\forall c, d, e, n$: $RL!(a, c, d, e, n) \Leftrightarrow RL!(b, \varphi c, \varphi d, \varphi e, n)$

(iv) $\forall c$: $Peg(c) \Leftrightarrow Peg(\varphi c)$

(v) $\forall c$: $Peg(c) \Rightarrow \varphi c \equiv c$

It's easy to check this is reflexive, symmetric, transitive, so therefor defines an equivalence relation.

Oops! This definition is too limiting. It rules out symmetric graph constructions like

~~cow~~ confusing it for

. This is undesirable. I only cop on to this on page 81.

20230617 - SGL Reductions - Intro

SGL so far seems to only be a passing curiosity, however it's most useful to model formal systems like logic and computations.

As motivation, we'd like to be able to model:

- Functional programs
- Imperative programs
- Turing machines
- Finite automata
- Logical systems
- Type theories

And importantly:

- Proofs about any system.

All in the same language, using the same reduction rules, where all intermediate states have a clear connection to intermediate states in their 'native' interpretation (although extra intermediate states are often needed).

The reason for this unification is that theorems and insights in one field are sometimes transferrable across to a seemingly unrelated field, bringing great insights with it.

20230621 - Maps and Projections in SGL

To get to reductions we've got to get to pattern rules, which means going through maps and projections.

We've already seen permutations used for defining identity. This was a meta-language object, meaning that it's a conventional mathematical object as opposed to, say, nodes in an SGL graph. It is in fact an axiom, as we shall see, that any SGL-like graph under a couple of conditions is ~~in fact~~ a semantic graph.

Definability Axiom (model theory)

Let S, P be sets with $P \subseteq S$.

Let M be a multiset of S triples with $M \subseteq S^3$.

Then (S, P, M) is a semantic graph iff:

(i) $\forall a, b \in S.\ \equiv(a, b) \Leftrightarrow a \equiv b$, and

(ii) $\forall a \in S.\ |\{(b, c, d)_{\times n} \in M : \exists p \subseteq M.$
$$(\forall (e, f, g) \in p.\ (e \equiv a \vee \exists (h, i, e) \in p))$$
$$\wedge\ (b, c, d) \in p\}| < \infty$$

Under the interpretation:

$Peg : S \rightarrow \mathbb{B}$

$Peg(a) := a \in P$

$L : S^3 \times \mathbb{N} \rightarrow \mathbb{B}$

$L(a, b, c, n) := \{(a, b, c)_{\times n}\} \subseteq M$

Additionally,

(iii) $\forall (a, b, c)_{\times n}.\ a \notin P$

This could be put better.

A proto-SGL-model $\mathcal{M}$ is a tuple $(\mathbb{S}, P, L)$ with $\mathbb{S}, P$ sets, $\mathbb{S} \supsetneqq P$ and

L is a multiset, and

$\forall \ell \in L.\ \ell \in (\mathbb{S} \setminus P) \times \mathbb{S}^2$.

The <u>interpretation of a model</u> $(\mathbb{S}, P, L)$ is:

$\mathrm{Peg}(a) = (a \in P)$

$L(a, b, c, n) = ((a, b, c) \in^n L)$ for $n \geq 1$

$\mathrm{Domain} = \mathbb{S}$

Then a proto-SGL-model $(\mathbb{S}, P, L)$ is a (full-) SGL-model iff the following conditions hold:

(i) $\forall a, b \in \mathbb{S}.\ a \equiv b \Leftrightarrow a \cong b$

(ii) $\forall a \in \mathbb{S}.\ (\exists n \in \mathbb{N},\ \varphi : [0..n] \to L.$

$\forall b, c, d \in \mathbb{S},\ i \in \mathbb{N}.$

$RL(a, b, c, d, i) \Leftrightarrow \exists j \in [0..n].$

$\varphi(j) = (b, c, d, i))$

(iii) $\cancel{\exists \varphi}$ $\cancel{\forall}$ $\exists \varphi : \mathbb{N} \leftrightarrow P$ bijection. i.e.

There are countibly many pegs.

Condition (i) we're already seen. It tells us we can't have two isomorphic but distinct nodes. This gives us a tangible way of testing equality ~~out~~ inside an interpretation.

Condition (ii) restricts us to finite link-sets. We don't want infinite reaches because then some nodes will be unrepresentable in a computer, and can make basic checking undecidable. To create infinite sets, simply create a decider.

Aside: Not every set can be represented this way. It would have to be fully-specifiable with finite information. So a set of random integers might not be decidable. And the set of subsets of integers would only be decidable on sets that can be represented. If a value couldn't be represented, then you couldn't even present it to the decider as there'd be no matching SGLnode.

Lemma: There are countably infinite many distinct semantic graphs.

Proof: P is countably infinite and $P \subseteq S$ so S must be at least infinite.

Let $S|_n = \{ s \in S : \cancel{R} |\{ l \in L : RL(s, l)\}| = n \}$ be the set of nodes that reach exactly n links, including duplicates.

Then, since each link has 3 nodes, there may only be at most $3n$ nodes reachable from each $s \in S|_n$. Now note that every node reachable from s must only reach a subset of the links that s reaches. This is by the transitivity of RL.

So $\forall a \in S.\ \exists m \in [0..n].\ Reach(s, a) \Rightarrow (a \in S|_m$
$\wedge \forall l \in L.\ RL(a, l) \Rightarrow RL(s, l))$

Now given $p \in [0..3n]$ and a set P of pegs, having size p:

Let $T = [p..3n) \times [0..3n)^2$

Then $|T| = (3n - p) \times (3n)^2 \leq (3n)^3$

Let $U = \{ V \subseteq^* T.\ |V| = n \}$ be the set of size n multi-subsets of T.

~~Then it's easy to see that $|U| \leq \frac{(3n)^{3n}}{n!}$~~
~~since that would be the numerator~~

From combinatorics, the number of ways of n items from a selection of at most $(3n)^3$ is $|U| \leq \binom{(3n)^3 + n - 1}{n}$.

For our purposes, it simply matters that this is a finite quantity, so take $u \in \mathbb{N}$ s.t. $|U| = u$.

Next, we want to show that every $s \in S|_n$ ~~s.t. to th~~ with p pegs is isomorphic to a node in a graph given by some $V \in U$.

Let $L_s = \{ l \in^* L : RL(s, l) \}$. Then $|L_s| = n$.

Let $R_s = \{ a \in S : R(s, a) \}$. Have $|R_s| \leq 3n$

Let $f : R_s \to [0..3n)$ be defined as such that ~~f~~ f maps all pegs onto $[0..p)$ and f is 1-to-1. Possible since $|R_s| \leq 3n$.

Next, let $\varphi: L_S^* \to T \times^* \mathbb{N}$ be a multiset permutation s.t. $\varphi(a,b,c,n) = (fa, gb, fc, n)$. Since a is not a peg, φ is well defined and isomorphic on some multi-subset $V \subseteq^* T$, so $|V| = n$. Hence

$\forall s \in \mathbb{S}|_n$ $\forall p$ ~~peg~~ set $\exists V \in U, v \in V.\ s \cong v$.

Thus ~~18~~ $|\mathbb{S}|_n| \le |P|^{3n} \cdot u$

Take $g: P \to \mathbb{N}$ injection

Let $\pi: \mathbb{N} \to \mathbb{N}$ map the natural m onto the m^{th} prime.

~~Given~~

Let $\Psi: P^{3n} \times u \to \mathbb{N}$ be defined as

$$\Psi(x, w) = \prod_{i=0}^{3n-1} (\pi i)^{g(x(i))} \cdot (\pi(3n))^w$$

Then Ψ is injective by unique prime factorization.

Thus $\forall n \in \mathbb{N}.\ |\mathbb{S}|_n| \le |\mathbb{N}|$

Take $\Phi_n: \mathbb{S}|_n \to \mathbb{N}$ inj

Then let $R: \mathbb{S} \to \mathbb{N}$ be

$$R(s) = 2^{hg(s)} \cdot 3^{\Phi_{hs}(s)}$$

Where $h: \mathbb{S} \to \mathbb{N}$ is defined as

$$h(s) = |\{l \in L : RL(s, l)\}|$$

So s is always in $\mathbb{S}|_{hs}$.

R is onto, therefore $|\mathbb{S}| \le |\mathbb{N}|$

Hence $|\mathbb{S}| = |\mathbb{N}|$ □

Hence call a semantic model 'complete' iff it contains all possible nodes.

Now we won't do very much more model theory, instead preferring to work on the interpretation.

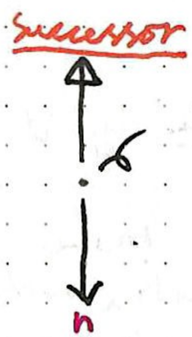
Definition of Formation

A formation is a decidable ~~subset~~ predicate on nodes along with an interpretation (written normally in plain English or mathematics) of each node that satisfies the predicate.

Let's start introducing formations by looking back to numerals mentioned in the introduction.

Formation: Successor

A successor is any node of the form



where n is any node at all. It could also be written:

$s \longrightarrow n$ or linearised to $s(n)$.

In all cases they represent the same node which has the meaning of having a value 1 greater than the value of n, wherever n actually has value. That last point is open-ended. The value a node has depends on its interpretation. You are free to pluck any unused peg, name it and assign any value to it or its

forms, provided they don't conflict with any other values/interpretation. You can create a lot of nonsense this way. What's s(successor)? It doesn't have a value because successor on its own doesn't really have a value. s(the-neighbour's-cat) doesn't have a value, unless the-neighbour's-cat somehow has a value that's a number. $s\circlearrowleft$ is an interesting node for a moment, but it doesn't have a value unless we give it one. It does resemble ω – the first transfinite ordinal, but the arithmetic is different. This is all to say that value and other interpretations are what you make them.

Now we want to define 'numeral', but we're going to have to go through an axiomatic method to ensure it is finite form within the theory. This is necessary because we will use the numeral definition to axiomatically ensure all other nodes have finite reach.

Axiom of Numerality

$$\cancel{\forall x:(\forall P\in\mathbb{P}_1:(\mathrm{Inductive}(P) \land (\forall Q\in\mathbb{P}_1: \mathrm{Inductive}(Q) \Rightarrow \forall y: P(y) \Rightarrow Q(y))) \Rightarrow P(x)) \Leftrightarrow \mathbb{N}(x)}$$

$$\forall x:(\forall P\in\mathbb{P}_1: \mathrm{Inductive}(P) \Rightarrow P(x)) \Leftrightarrow \mathbb{N}(x)$$

That is: $\mathbb{N}(x)$ holds iff $P(x)$ is true for all inductive predicates P. You could think of $\mathbb{P}_1$ as the set of all 1-argument predicates, but as a finitist, I prefer to think of $\mathbb{P}_1$ as a type and $P\in\mathbb{P}_1$ as a type judgement. So it could also be written $P:\mathbb{P}_1$. Next Inductive needs to be defined. The intuition is that a predicate is inductive if it 'contains' $\emptyset$ and every 'element' also ~~that~~ has its successor

'contained'.

Definition of Inductive

$$\forall P:\mathbb{P}_1.\ \mathrm{Inductive}(P) \Leftrightarrow (P(\emptyset) \land \forall x.\ P(x) \Rightarrow P(s(x)))$$

So $\mathbb{N}$ is the least predicate that holds for $\emptyset$ and all its successors.

Axiom of potentiality

$\mathrm{Inductive}(\mathbb{N})$

That's it. What this asserts is that there is an inductive set. This is required as there could be a model that only goes up to some natural n. However, it could also be expressed by asserting that every node has a successor.

Axiom of Extension

A predicate $P : \mathbb{P}_1$ is fully-extended exactly when $\forall a, b, c . (P(a) \land L(a,b,c)) \Rightarrow (P(b) \land P(c))$. Then write $\text{FullyExtended}(P)$. We may use the same trick as we did for $\mathbb{N}$, namely to define a predicate as the minimal of all possible predicates satisfying a second-order condition.

$$\forall a, b : (\forall P : \mathbb{P}_1 . (\text{FullyExtended}(P) \land P(a)) \Rightarrow P(b))$$
$$\Leftrightarrow \underline{\text{Reach}}(a, b)$$

This makes the definition of RL (reach-link) and RL! (exact reach-link) easy. But I haven't formally defined L and L!, so let's do them all together.

Definition of L, L!, RL, and RL!

L is a 4-argument predicate and is one of the two atomic predicates along with Peg that give meaning to an SGL theory.

$L(a, b, c, n)$ holds whenever $\mathbb{N}(n)$, $n \not\equiv \underline{\emptyset}$, and there are at least n instances of the link $a \xrightarrow{b} c$ from a, where a is not a peg.

$$L(a, b, c, n) \Rightarrow \mathbb{N}(n) \land \neg(n \equiv \underline{\emptyset}) \land \neg \text{Peg}(a)$$

$n \not\equiv \underline{\emptyset}$ as this results in $\{(a, b, c, n) \mid L(a, b, c, n)\}$ to have a corresponding element in the model's multiset Λ. But back to the theory...

$$\forall a, b, c, n, m : (L(a, b, c, n) \land \text{Reach}(n, m) \land \neg(m \equiv \underline{\emptyset}))$$
$$\Rightarrow L(a, b, c, m)$$

This says that if $L(a, b, c, n)$ then $L(a, b, c, m)$ for any $\underline{\emptyset} < m \leq n$.

$$L!(a, b, c, n) \Leftrightarrow L(a, b, c, n) \land \neg L(a, b, c, s(n))$$

That is when there are exactly n links.

$$RL(a, b, c, d, n) \Leftrightarrow \text{Reach}(a, b) \land L(b, c, d, n)$$

$$RL!(a, b, c, d, n) \Leftrightarrow \text{Reach}(a, b) \land L!(b, c, d, n)$$

These last two are just short-hand and should be fairly self-explanitory.

Formation: Numeral

We have already defined $\underline{0}$, successor, and $\mathbb{N}$. This formation ties them together. A node a is a numeral iff $\mathbb{N}(a)$. Since $\mathbb{N}$ is Inductive, we know that $\mathbb{N}(\underline{0})$ and that $\mathbb{N}$ is closed under successor, $\forall x: \mathbb{N}(x) \Rightarrow \mathbb{N}(s(x))$. We'd like to prove that $\forall x. \mathbb{N}(x) \Rightarrow$ 'x is a finite chain of successors ending at $\underline{0}$' but defining 'finite' depends on $\mathbb{N}$, making the process circular. So we will show it from a model-theoretic view, writing the model side in red. We want to show the following:

~~$\forall x \in S. \exists n \in N; \varphi:[0..n] \Leftrightarrow$~~

Lemma: $\forall x \in S. \mathbb{N}(x) \Leftrightarrow (\exists n \in N; \varphi:[0..n] \to S. \varphi(n)=x$

$\wedge\ \varphi(0)=\underline{0}\ \wedge \forall m \in N. m < n \Rightarrow (\varphi(m+1)$ is a successor $\wedge\ L(\varphi(m+1), \underline{d}, \varphi(m)))$

This states that every natural number is a finite chain of successors specified by φ.

Prog (in model theory):

First we define a predicate P as follows:

$$\forall x. P(x) \Leftrightarrow \exists n \in N; \varphi:[0..n] \to S.$$
$$\varphi(n) = x$$
$$\wedge\ \varphi(0) = \underline{0}$$
$$\wedge\ \forall m \in N. m < n \Rightarrow ($$
$$\varphi(m+1) = s(\varphi(m)))$$

Then $P(\underline{0})$ as

$n = 0$, $\varphi: \{0\} \to S$ with $\varphi(0) = 0$ defines φ, and there is no $m \in N$ s.t. $m < n$ so the last clause is vacuously true.

Given x, suppose $P(x)$ holds, then we want to show that $P(s(x))$ also holds to show that Inductive(P).

Take $n \in N; \varphi:[0..n] \to S$ s.t. $\varphi(n) = x$, $\varphi(0) = \underline{0}$, and $\forall m \in N. m < n \Rightarrow (\varphi(m+1) = s(\varphi(m)))$.

Now let $n' = n+1$, and let $\varphi' : [0..n'] \to S$ be defined as:

$$\varphi'(m) = \begin{cases} \varphi(m) & \text{if } m \leq n \\ s(\varphi(n)) & \text{if } m \leq n' \end{cases}$$

Then $\varphi'(n') = s(\varphi(n)) = s(x)$ and $\varphi'(0) = \varphi(0) = 0$, which are two of the three conditions for $P(s(x))$.

Given $m \in \mathbb{N}$ with $m < n'$,

Then $m \leq n$

Suppose $m < n$, then we already know that $\varphi(m+1) = s(\varphi(m))$.

So it remains to show that ~~$\varphi(m+1)$~~ $\varphi(n+1) = s(\varphi(n))$. But this is by definition.

Ergo $P(s(x))$.

Therefore $P(0) \wedge \forall x. P(x) \Rightarrow P(s(x))$.

This precisely says Inductive (P).

Now we'd like to use the definit of $\mathbb{N}$, but we

don't actually know that $P \in \mathbb{P}_1$. That's because it's a model theoretic predicate. Let's define a model theoretic

$\underline{\mathbb{N}}$: $\forall x \in S$, $\underline{\mathbb{N}}(x) \Leftrightarrow \forall Q$ predicate. Inductive $(Q) \Rightarrow Q(x)$

Suppose $\underline{\mathbb{N}}(x)$

Then $\forall Q$ predicate. Inductive$(Q) \Rightarrow Q(x)$

So in particular Inductive$(P) \Rightarrow P(x)$

Thus $P(x)$

So this establishes $\underline{\mathbb{N}}(x) \Rightarrow P(x)$

Next suppose $P(x)$

Given Q predicate s.t. Inductive(Q)

Suppose for a contradiction that $Q(x)$ is false.

Take $n \in \mathbb{N}$; $\varphi : [0..n] \to S$ s.t. the clauses of $P(x)$ hold.

Then there must be an $m \in$ ~~$[0..$~~ $[0..n)$ s.t.

$Q(\varphi(m))$ but $\neg Q(\varphi(m+1))$ since

$Q(\varphi(0)) = Q(0)$ holds since Q is inductive.

But $Q(\varphi(n)) = Q(x)$ doesn't hold.

So $Q(\varphi(m))$ but $\neg Q(s(\varphi(m)))$ ※ Q closed under successor.

Hence $Q(x)$ holds.

This gives us the other direction: $P(x) \Rightarrow \mathbb{N}(x)$ so $P(x) \Leftrightarrow \mathbb{N}(x)$. It remains to show $\mathbb{N}(x) \Leftrightarrow \mathbb{N}(x)$.

Have $\mathbb{N}(x) \Rightarrow \mathbb{N}(x)$ as all object predicates ($\mathbb{P}_1$) are predicates in the meta-language.

Finally we want to show that $\mathbb{N}$ is no 'smaller ~~bigger~~' than $\mathbb{N}$.

~~Suppose $\mathbb{N}(x)$ $\mathbb{N}(x)$~~

Suppose for a contradiction that there is an $x \in S$ such that $\mathbb{N}(x)$ but not $\mathbb{N}(x)$

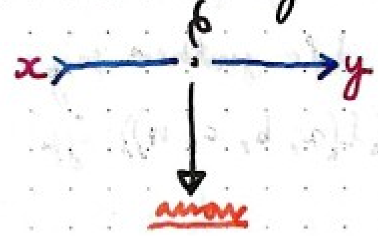
Then take predicate P s.t. Inductive(P) but $\neg P(x)$

...? I think it's not provable!! See the Löwenheim-Skolem theorem.

So turns out I do need full second order to scare away the non-standard models. Good to know.

Formation: Arrow (>→)

An arrow is any node of the form



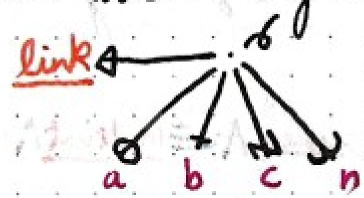
where —< is arrow-tail and —> is arrow-head for any nodes x, y.

It's often drawn $x \succ\!\!\longrightarrow y$ and color-coded based on what map/projection they're part of.

What it means depends on context, but usually means that x is mapped to y in some sense.

Formation: Link-node

A link-node is any node of the form



where in this case the ordered-links have been reused, standing in for 0: object, 1: role, 2: target, 3: duplicity. This is a common trick we'll use over again. It saves us inventing new symbols for every formation, even if we do

mean to pluck new pegs. Write $\ell(a, b, c, n)$.
We say this node is a link node iff it is both of the above form, and $\mathbb{N}(n)$. We further say $\ell(a, b, c, n)$ holds, or exists, and write $L(\ell(a, b, c, n))$ iff $L(a, b, c, n)$.
The appeal of doing so is simply that it lets us consider L as a 1-place predicate rather than a 4-place predicate, which can make it easier to work with as we'll see.

Formation: Multiset and Set

A multiset is any node n in which the following hold:

(i) $L!(n, \underline{\text{super}}, \underline{\text{multiset}}, 1)$

(ii) $\forall b, c, m.\ L!(n, b, c, m) \Rightarrow (b \equiv \underline{\text{super}} \land c \equiv \underline{\text{multiset}} \land m \equiv 1)$
$\lor\ b \equiv \underline{\text{element}}$

And similarly for set:

(i) $L!(n, \underline{\text{super}}, \underline{\text{set}}, 1)$

(ii) $\forall b, c, m.\ L!(n, b, c, m) \Rightarrow (b \equiv \underline{\text{super}} \land c \equiv \underline{\text{set}} \land m \equiv 1)$
$\lor\ (b \equiv \underline{\text{element}} \land m \equiv 1)$

We say, given a type T that a node n is of type $\text{Multiset}(T)$ iff n is a multiset and additionally $\forall c.\ L(n, \underline{\text{element}}, c) \Rightarrow T(c)$. And likewise for $\text{Set}(T)$.
Both set and multiset represent an unordered collection of stuff.
The predicate form of a set n is written $\overset{\in}{\cancel{P}}(n)$ and is defined as:

$$\forall c.\ \underset{\in}{\cancel{P}}(n)(c) \Leftrightarrow L(n, \underline{\text{element}}, c)$$

We may write $n \equiv \{a_1, \ldots, a_m\}$ for the set containing $a_1, \ldots, a_m$ and nothing else. We will see that the set/multiset constructions as well as all other formations must necessarily be finite.
If we have both arguments then $\in$ may be written infix in the familiar form:

$$c \in n$$

We write $n \subset m$ if n and m are both sets/multisets and $\forall b, c, i.\ L(n, b, c, i) \Rightarrow L(m, b, c, i)$

Formation: Map

A map is simply a set of arrows.

We say a map m is proper iff

$$\forall a,b,c. ((a \twoheadrightarrow b) \in m \land (a \twoheadrightarrow c) \in m) \Rightarrow b \equiv c$$

That is: m may not link any node a to more than one destination. That's the only requirement upon being a set of arrows. You'll see a clear connection between proper maps in SGL and maps in set theory, which must be a set of pairs where each left-hand element may be paired with at most one right-hand element.

Function: domain & range

These are colored green to indicate that it is a symbol in the meta-theory.

$$\begin{aligned}\forall x. \exists! y.\ \mathrm{domain}(x) \equiv y \land (&\\ &(\neg \mathrm{Map}(x) \land y \equiv \mathrm{exception}(\text{not-of-type}, \text{map}, x))\\ \lor\ (&\mathrm{Map}(x) \land \mathrm{Set}(y) \land\\ &\forall u,v. (u \twoheadrightarrow v) \in x \Leftrightarrow u \in y))\end{aligned}$$

Where exception, not-of-type, and map are newly recruited pegs and the exception formation is defined below.

range is defined almost identically:

$$\begin{aligned}\forall x. \exists! y.\ \mathrm{range}(x) \equiv y \land (&\\ &(\neg \mathrm{Map}(x) \land y \equiv \mathrm{exception}(\text{not-of-type}, \text{map}, x))\\ \lor\ (&\mathrm{Map}(x) \land \mathrm{Set}(y) \land\\ &\forall u,v. (u \twoheadrightarrow v) \in x \Leftrightarrow \cancel{u}\, v \in y))\end{aligned}$$

Then both domain and range are well-defined as either $\mathrm{Map}(x)$ or $\neg \mathrm{Map}(x)$ so ~~exact~~ at most one of the two disjunctive clauses may be true at one time. If $\neg \mathrm{Map}(x)$ then y exists and is unique since it is a result of a constructor that reaches no other variables but x. On the other hand, if $\mathrm{Map}(x)$ does hold, take y_1, y_2 s.t. they're both sets and the light-green '$\forall$' condition holds. Then given $\forall u,v$, we have $(u \twoheadrightarrow v) \in x \Leftrightarrow u \in y_1$

and $(u \twoheadrightarrow v) \in x \Leftrightarrow u \in y_2$.

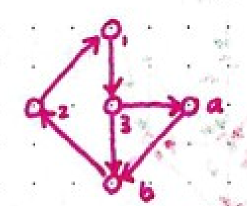
20230713 Canonical Definitions

What we've been doing up to now is trying to bootstrap (construct from basic parts) the formations, predicates and functions ~~axioms~~ we need to even formulate the SGL axioms from the theory side. To do this, we need to make sure any function we define has exactly one output for each input. This is held up on the invalid-input case by outputting an exception (which itself needs to be validated), however when constructing nodes by defining their outbound links, there's two hang-ups that may prevent them from being unique:

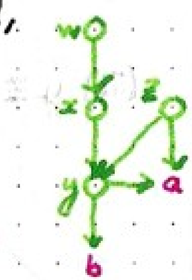
1. Unintended cycles - As seen overleaf, link targets may happen to end up being the output object (or an intermediate object node).
2. Node factorization - This is where the existance of an outer-cycle causes ambiguity about (what is typically considered) irrelevant graph features, such as the duplication of nodes with the same outgoing links.

Let's make this concrete by way of example. We've seen a very basic unintended cycle in the last section. The following, more involved example, has been simplified to show only the structure of the graph. As such, it is called a structural diagram.

Example: Consider the graph below. We will consider it fully defined. Take a and b labelled.



Next, the graph on the right is only link-defined, meaning that w, x, y and z are variables that vary together over all node 'groups' that have exactly the same number of outward links with the same roles (ignored in this case) and targets.

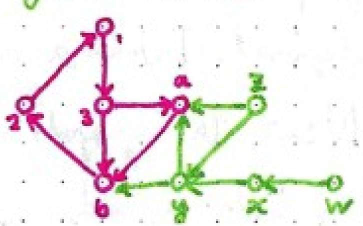


Then there's a certain ambiguity to x, y, z and w, hinted at by the way they're drawn. In particular:

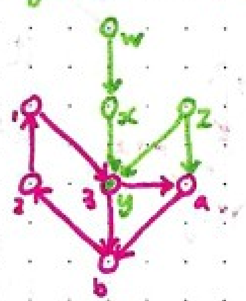
- y and 3 point to both a and b, so it's possible that they could be equivalent.
- If $y \equiv 3$ then x and 1 would both point only to 3. This makes them both candidates for aliasing, defined below.

Further checking reveals no more ambiguities. This paints 3 possibilities:

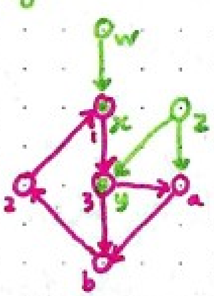
(i) $y \not\equiv 3 \wedge x \not\equiv 1$:



(ii) $y \equiv 3 \wedge x \not\equiv 1$:



(iii) $y \equiv 3 \wedge x \equiv 1$



Definition: Aliasing

An 'alias' simply means another name for something. In SGL it means just that, a node that's referred to by two different variables.

In the last example, the only reason the ambiguity occurred is that $b \to 2 \to 1 \to 3$ forms a cycle. If the link from 2 to 1 didn't exist then the distinction between y and 3 would no longer hold as there's no non-trivial path from y to 3 any more. That's the crux of unintended cycles: If a set of variables ($\{w, x, y, z\}$) are link-defined ~~then~~ and a node (y) can reach a node (3) that is link-equivalent to that node, then the ambiguity is suddenly present; y could refer to itself or it could refer to the distinct node 3. If instead y couldn't reach 3 then the distinction would no-longer be relevant to the definition of y.

Let's lay down a couple definitions.

Definition: Link-defined

~~Fix V to be a set of variables~~.

~~Say a predicate~~

Let P be some n place predicate.

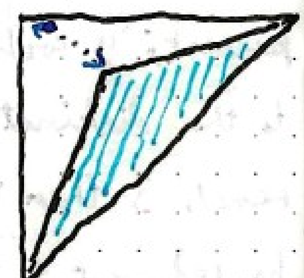
Say P is directly link-defined iff

there is a sequence $\bar{x}$ of n nodes s.t. $P(\bar{x}) \wedge$

for every $\bar{y}$ sequence of n nodes,

if $P(\bar{y})$ then there is a homomorphism from the reach-graph of $\bar{x}$ to the reach-graph of $\bar{y}$.

In this $\bar{x}$ is the minimally aliased solution. In the last example, solution (i) was the minimally aliased solution. Homomorphisms allow variables ~~nodes~~ ~~to~~ 'fold' nodes to 'fold' onto one another if they have the same link-set.



Exactly the property we want.

We will define homomorphisms shortly.

If Φ is a formula (meta-formula) with free variables $\bar{V}$ and there is predicate P that is directly link defined s.t. $\forall \bar{V}. P(\bar{V}) \Leftrightarrow \Phi$, then we can say for any v in $\bar{V}$ that v is (indirectly) link-defined.

Definition: Homomorphism (incorrect)

~~A map φ from {~~

A map φ is a homomorphism iff

~~there is~~

(i) 'φ is total'

$$\forall a, a', b.\ \cancel{\varphi(a, a')}\ \varphi(a) \equiv a' \wedge Reach(a,b) \Rightarrow \exists b'.\ \varphi(b) \equiv b'$$

(ii) $\forall a, a', b'.\ \varphi(a) \equiv a' \wedge Reach(a', b') \Rightarrow \exists b.\ \varphi(b) \equiv b'$

(iii) $\forall a, b, n, c.\ (\exists a'.\ \varphi(a) \equiv a') \Rightarrow (L(a,b,c,n) \Leftrightarrow L(\varphi a, \varphi b, \varphi c, n))$

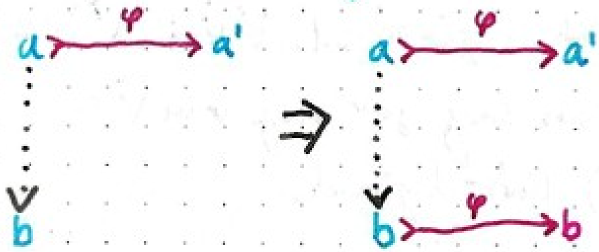
(iv) $\forall a, b', c'.\ ((\exists a'.\ \varphi(a) \equiv a') \wedge L(\varphi a, b', c', n)) \Rightarrow \exists b, c.\ \varphi b \equiv b' \wedge \varphi c \equiv c' \wedge L(a,b,c,n)$

(v) $\forall a, a'.\ \cancel{Peg}\ \varphi(a) \equiv a' \Rightarrow$
$(Peg(a) \Leftrightarrow Peg(a'))$

(vi) $\forall a, a'.(\varphi(a) \equiv a' \wedge Peg(a)) \Rightarrow a \equiv a'$

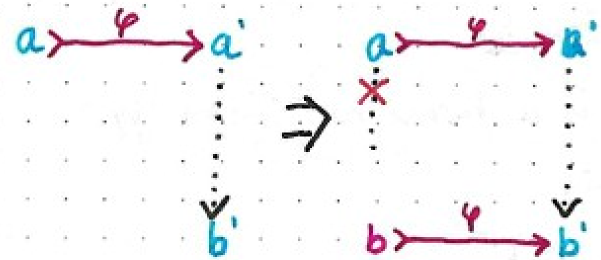
That is:

(i)



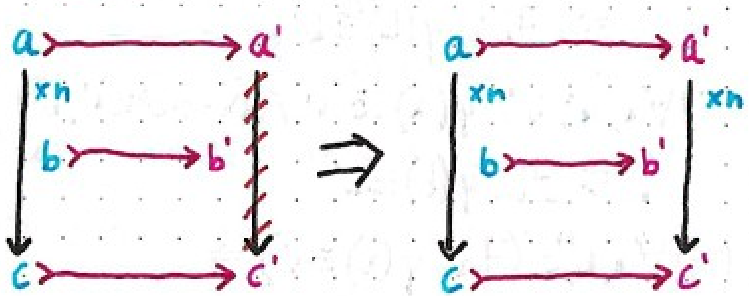
Or domain(φ) is closed under Reach

(ii)

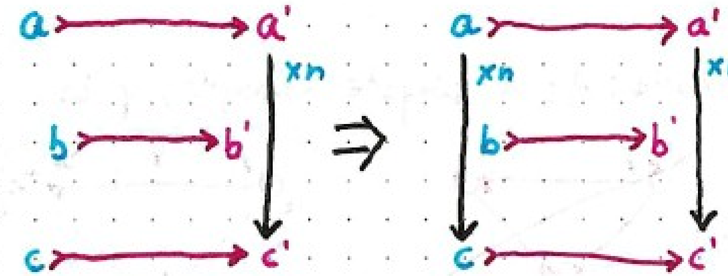


Or range(φ) is closed under Reach

(iii)

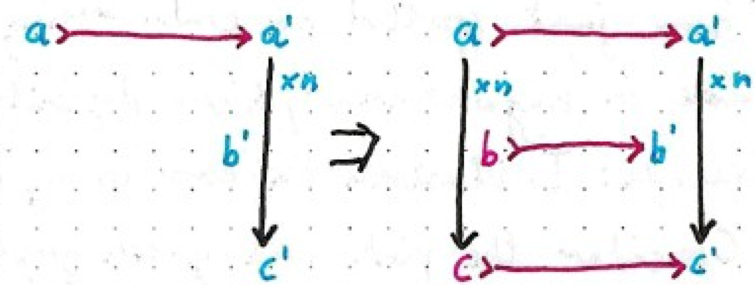


Or every link in the domain can be projected through φ



Or every projected link in the range is projected from some link in the domain.

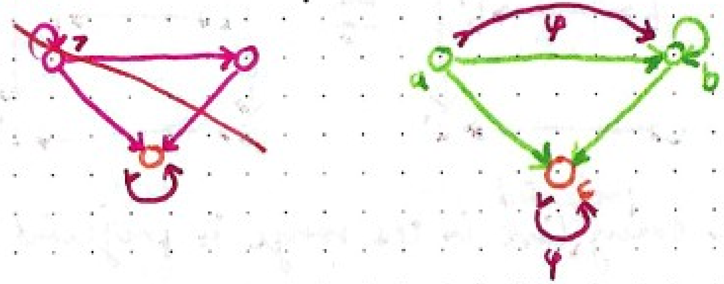
(iv)



Or every link in the range can be projected from a link in the domain.

(v) & (vi) say that every peg must be projected onto itself and only pegs may project onto pegs. This grounds φ by requiring it to only project between nodes with 'similar' meanings.

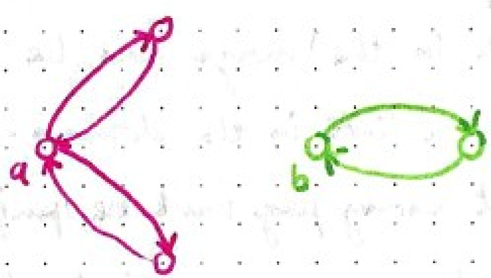
Example: As a simple example, φ in



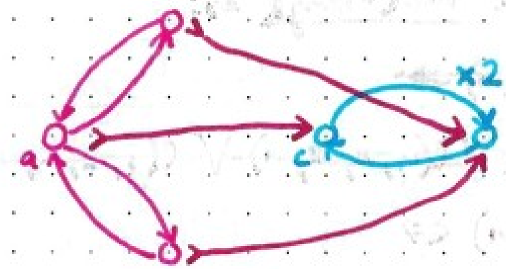
is a homomorphism.

I have just spotted an embarrassing error I made in my homomorphism definition.

Example: To illustrate the error in my last definition, Consider the pink and green graphs below



my definition stated that a should be homomorphic to b. This must be erroneous because a has two links out while b has just one. Instead it only makes sense to say that a is homomorphic to the following c:



So how can we change the definition to make this work? The problem appears to be with (iii) and (iv). They don't preserve multiplicity of outgoing links because they allow multiple links to be smoothed together. We don't want to allow that which requires us tracking every single link. To do that we need to add to the definition an idea of link-permutations.

Definitions: Link-map

A map m is a link-map iff all elements of the domain and range are link-nodes.

A link-map m is situated iff

$$\forall l_1, l_2 . (l_1 \mapsto l_2) \in m \Rightarrow L(l_1) \wedge L(l_2)$$

Define link-domain and link-range simply as:

link-domain = domain   link-range = range

Define node-domain and node-range as:

$$\cancel{\forall x.\ x \in \text{node-domain}(m) \Leftrightarrow}$$

$$\cancel{\exists y, z, n.\ \ell(x,y,z,n) \vee \ell(y,x,z,n) \vee \ell(y,z,x,n)}$$

$$\forall x.\ x \in \text{node-range}(m) \Leftrightarrow$$

$$\begin{aligned}
&\exists y, z, n, \ell.(\\
&\quad \ell \equiv \ell(x,y,z,n)\\
&\vee\ \ell \equiv \ell(y,x,z,n)\\
&\vee\ \ell \equiv \ell(y,z,x,n)) \Rightarrow \ell \in \text{range}(m)
\end{aligned}$$

(And analogously for node-domain)

A link-map is closed iff

$$\forall x.\ x \in \text{node-domain}(m)\ \cancel{\Leftrightarrow} \Rightarrow$$

$$\forall a,b,c,n.\ RL(x,a,b,c,n) \Rightarrow$$

$$\ell(a,b,c,n) \in \text{link-domain}(m)$$

$\wedge$ (and likewise for range)

A link-map m is said to follow a node-map n iff

$$\forall a,a',b,b',c,c',i,j.\ m(\ell(a,b,c,i)) \equiv \ell(a',b',c',j) \Rightarrow$$

$$na \equiv a' \wedge nb \equiv b' \wedge nc \equiv c'$$

Definition: Injection

A map m is an injection iff

$$\forall a,b,c.\ m(a) \equiv c \wedge m(b) \equiv c \Rightarrow a \equiv b$$

Definition: Out-link injective

A link-map m is out-link injective iff

$$\cancel{\forall a,b,c,i.\ a \in \text{node-domain}(m) \wedge L(a,b,c,i) \Rightarrow}$$

$$\forall a, b_1, b_2, c_1, c_2, i_1, i_2.$$

$$a \in \text{node-domain}(m) \wedge m(a,b_1,c_1,i_1) \equiv m(a,b_2,c_2,i_2)$$

$$\Rightarrow b_1 \equiv b_2 \wedge c_1 \equiv c_2 \wedge i_1 \equiv i_2$$

Which says that if a node a is fixed then the links with a as their subject ~~cause~~ make m an injection when restricted to this set.

~~Definition: Out-link surjective~~

~~A link-map m~~

Definition: Homomorphism

A map φ is a homomorphism iff there is a link-map ϕ such that the following hold:

- ϕ follows φ
- ϕ is closed (on both the domain and range)
- ϕ is out-link ~~injective~~ bijective on φ
- ϕ is situated
- ~~ϕ is transmitted by φ~~ ϕ and φ are proper maps
- φ respects pegs

Then (i) and (ii) of the old definition are covered by closure. The forward case of (iii) is covered by transmission, and the backward case again by closure but also 'situation', every reachable link in the ~~domain~~ range must be mapped to by ϕ which must follow φ. (iv) is analogous to out-link surjection. (v) & (vi) are covered by the last point.

Definition: Out-link surjection & bijection

A link-map ϕ is an out-link surjection on a map φ iff

$$\forall a, a', b', c', j.\ \varphi a \equiv a' \land L(a', b', c', j) \Rightarrow$$
$$\exists b, c, i.\ \phi(\ell(a, b, c, i)) \equiv \ell(a', b', c', j)$$

This can be changed to the definition of out-link bijection simply by changing the '$\exists$' to 'exists a unique' '$\exists!$'. Then all situated ~~links~~-maps ϕ that follows a map φ are out-link bijective iff they are out-link surjective and injective.

Proof:

Given φ and ϕ that satisfy the premises,

Suppose ϕ is an out-link bijection on φ.

Then ϕ is automatically an " surjection on φ as '$\exists! \Rightarrow \exists$'

Given $a, b_1, b_2, c_1, c_2, i_1, i_2$;

Suppose $a \in$ node-domain(ϕ),

and $\phi(a, b_1, c_1, i_1) \equiv \phi(a, b_2, c_2, i_2)$

Since ϕ ~~follows φ and~~ is situated,

$L(\phi(a, b_1, c_1, i_1))$

Since ϕ follows φ, take j s.t.

$$\phi(a, b_1, c_1, i_1) \equiv \ell(\varphi a, \varphi b_1, \varphi c_1, j)$$

Then it immediately follows from uniqueness in bijection that $b_1 \equiv b_1 \wedge c_1 \equiv c_2 \wedge i_1 \equiv i_2$

Therefore ϕ is an injection

Suppose instead that ϕ is an injection and a surjection on φ

Given a, a', b', c', j s.t. $\varphi a \equiv a'$ and

$L(a', b', c', j)$

By surjection,

$$\exists b, c, i.\ \phi(a, b, c, i) \equiv \ell(a', b', c', j)$$

Given $b_1, b_2, c_1, c_2, i_1, i_2$ s.t.

$$\phi(a, b_1, c_1, i_1) \equiv \ell(a', b', c', j)$$
$$\phi(a, b_2, c_2, i_2) \equiv \ell(a', b', c', j)$$

Have $a \in \text{node-domain}(\phi)$

and $\phi(a, b_1, c_1, i_1) \equiv \phi(a, b_2, c_2, i_2)$

Therefore $b_1 \equiv b_2 \wedge c_1 \equiv c_2 \wedge i_1 \equiv i_2$

by injectivity

Hence ϕ is a bijection

Hence the result □

~~Definition: Transmission~~

~~A link-map ϕ is transmitted by φ iff~~

~~$\forall a, b, c, i. (\exists j. \ell(a, b, c, i) \in \text{link-domain}(\phi) \Rightarrow$~~

~~$\phi(a, b, c, i) \equiv \ell(\varphi a, \varphi b, \varphi c, j))$~~

~~That is every arrow in the link map~~

(Actually this is redundant with follow)

Definition: Isomorphism

A map φ is an ~~permutation~~ isomorphism iff it is a homomorphism and a bijection.

This is a simple definition but let's check it has the properties we want.

Claim: If φ is an iso-morphism then ~~there~~ is any ~~any~~ corresponding link-map ϕ ~~that~~ is a bijection.

Proof: Since ϕ is an out-link bijection on φ,

$$\forall a, b', c', j. (a \in \text{node-domain}(\phi) \wedge \cancel{L\,a}$$
$$\wedge L(\varphi a, b', c', j)) \Rightarrow \exists! b, c, i \ldots$$

Given $a', b', c', j, a_1, a_2, b_1, b_2, c_1, c_2, i_1, i_2$,

s.t. $\varphi(a_1, b_1, c_1, i_1) \equiv \varphi(a_2, b_2, c_2, i_2) \equiv \ell(a', b', c', j)$

Since ϕ follows φ,

$$a' \equiv \varphi a_1 \equiv \varphi a_2 \land b' \equiv \varphi b_1 \equiv \varphi b_2 \land$$
$$c' \equiv \varphi c_1 \equiv \varphi c_2$$

Since φ is a bijection,

$$a_1 \equiv a_2 \land b_1 \equiv b_2 \land c_1 \equiv c_2$$

Since ϕ is situated, and since it's closed,

$$\forall a, b, c, i.\ \ell(a, b, c, i) \in \text{link-domain}(\phi) \Rightarrow$$
$$\forall j.\ \text{Reach}(i, j) \Rightarrow \land\, \mathbb{N}(j) \Rightarrow$$
$$\ell(a, b, c, j) \in \text{link-domain}(\phi)$$

And similar holds for link-range.

$(\text{Reach}(x, y) \Leftrightarrow x \geq y$ when $\mathbb{N}(x) \land \mathbb{N}(y))$

Now let ϕ be defined as follows:

$$\cancel{\forall a, b', c', j.\ L(\varphi a, b', c', j)} \land L(a, b, c, i)$$
$$\forall a, b, c, i.\ a \in \cancel{\text{node-}}\text{domain}(\varphi) \Rightarrow$$
$$\phi(a, b, c, i) \equiv \ell(\varphi a, \varphi b, \varphi c, i)$$

and $\forall a, b, c, i.\ a \notin \text{domain}(\varphi) \lor \neg L(a, b, c, i)$

$$\Rightarrow \ell(\varphi a, \varphi b, \varphi c, i) \notin \text{domain}(\phi)$$

TODO: Cleanup and finish

Lemma: $\forall a. \exists \varphi.\ \text{Isomorphism}(\varphi, a, a)$

Proof: Given a let S be a set s.t.

$$\forall b.\ \text{Reach}(a, b) \Leftrightarrow b \in S$$

Let φ be a map with domain S s.t.

$$\forall b.\ b \in S \Rightarrow \varphi(b) \equiv b$$

Let ϕ have domain

$$\forall b, c, d, n.\ \ell(b, c, d, n) \in \text{link-domain}(\phi) \Leftrightarrow$$
$$\text{RL}(a, b, c, d, n)$$

and be defined as

$$\forall b, c, d, n.\ \text{RL}(a, b, c, d, n) \Rightarrow$$
$$\phi(b, c, d, n) \equiv \ell(b, c, d, n)$$

Then it can be seen that ϕ follows φ.

ϕ and φ are closed since RL and Reach are closed.

ϕ and φ are bijections as all identity maps are injective and out-link surjective as ϕ follows φ which is bijective.

ϕ and φ are also clearly proper and situated

and finally respect pegs.

Hence it's an isomorphism □

Definition: Inverse

Let φ be a bijective map with domain S and range T.

Write φ^- for the inverse of φ formed by replacing each arrow $(a \rightarrow b)$ in φ with $b \rightarrow a$.

Then φ^- is proper ~~and~~ as φ is injective and we can show $\text{domain}(\varphi^-) \equiv T$ and $\text{range}(\varphi^-) \equiv S$.

Claim: $\forall a, b, \varphi.\ \text{Isomorphism}(\varphi, a, b) \Rightarrow$
$$\text{Isomorphism}(\varphi^-, b, a)$$

Proof:

Let a, b, φ be s.t. φ is an isomorphism between a's reachset and b's.

Let ϕ be a link-map corresponding to φ

Know that ϕ is a bijection.

Since ϕ follows φ and they're both bijections,

$$\forall a, a', b, b', c, c', i, i'.\ \phi^-(a', b', c', i') = \ell(a, b, c, i)$$
$$\Rightarrow \phi(a, b, c, i) = \ell(a', b', c', i') \text{ as } \phi \text{ bij}$$
$$\Rightarrow a' = \varphi a,\ b' = \varphi b,\ c' = \varphi c \text{ as } \phi \text{ follows } \varphi$$

$\Rightarrow \varphi^{-}a' = a,\ \varphi^{-}b' = b,\ \varphi^{-}c' = c$

as φ bij

So ϕ^{-} follows φ^{-}

ϕ^{-} is closed and situated as ϕ is.

They're both proper maps as they're inverse of bijections.

ϕ^{-} is out-link bijective, following from it being bijective.

And φ^{-} respect pegs as that property is invertable.

Hence φ^{-} is also an isomorphism □

This gives us two out of the three properties we need to check to ensure a relation is an 'equality relation'. For the last one, we want isomorphism to be transitive. We'll prove the transitivity of homomorphisms more generally. First, we would like to define composition

Definition: Composition

Given a map m and n and sets S, T, U s.t. the domain of m is S, the range of m and the domain of n is T and the range of n is U;
The composition is written $m \circ n$ and is defined to be the unique map s.t.

- $\forall a, b, c.\ (a \rightarrowtail c) \in m \circ n \Leftrightarrow$
  $\exists b.\ (a \rightarrowtail b) \in m \land (b \rightarrowtail c) \in n$
- and $\neg \mathrm{Reach}^{+}(m \circ n, m \circ n)$

That second clause may seem to come out of nowhere but it's vital to all definitions, to ensure uniqueness. We'll be able to prove this once we've reached the axiom of definability.

The theorems from set theory translate directly onto here so we will state without proof:

- If m and n are both proper then so is $m \circ n$.
- $m \circ n$ has domain S and range U.
- If m and n are both bijections then so is $m \circ n$.

Lemma: If S, T, U are sets; $\varphi_1 : S \to T$ and $\varphi_2 : T \to U$ are homomorphisms then $\varphi_1 \circ \varphi_2$ is also a homomorphism.

Proof: Take ϕ_1, ϕ_2 link-maps for φ_1, φ_2 resp.

WTS $\phi_1 \circ \phi_2$ is a link-map for $\varphi_1 \circ \varphi_2$.

Since $\phi_1 \circ \phi_2$ inherits the domain from ϕ_1 and range from ϕ_2, we already know that $\phi_1 \circ \phi_2$ is closed and situated.

We also know $\phi_1 \circ \phi_2$ is proper as ϕ_1 and ϕ_2 are.

(pegs): Given a peg a; if $a \in S$ then $(a \mapsto a) \in \phi_1$ since ϕ_1 respects pegs and also $(a \mapsto a) \in \phi_2$ likewise. Thus $(\phi_1 \circ \phi_2)(a) \equiv a$.

Given b, b is a peg iff $\varphi_1 b$ is a peg iff $\varphi_1(\varphi_2(b)) \equiv (\varphi_1 \circ \varphi_2) b$ is peg

(follows): Given $a, a'', b, b'', c, c'', n, n''$.

Suppose $(\varphi_1 \circ \varphi_2)(a, b, c, n) \equiv \ell(a'', b'', c'', n'')$

Take a', b', c', n' s.t. $\varphi_2(a', b', c', n') \equiv \ell(a'', b'', c'', n'')$ and $\varphi_1(a, b, c, n) \equiv \ell(a', b', c', n')$

Then it follows that: $\varphi_1 a \equiv a'$, $\varphi_1 b \equiv b'$ and $\varphi_1 c \equiv c'$; and $\varphi_2 a' \equiv a''$, $\varphi_2 b' \equiv b''$ and $\varphi_2 c' \equiv c''$

So simply $(\varphi_1 \circ \varphi_2) a \equiv a''$, $(\varphi_1 \circ \varphi_2) b \equiv b''$ and $(\varphi_1 \circ \varphi_2) c \equiv c''$

(out-link bijection): Given a, a'', b'', c'', j s.t. $(\varphi_1 \circ \varphi_2) a \equiv a''$ and $L(a'', b'', c'', j)$. Take a' s.t. $\varphi_1 a \equiv a'$ and $\varphi_2 a' \equiv a''$ and $\cancel{a}$ b', c' s.t. $\phi_2(a', b', c', j) \equiv \ell(a'', b'', c'', j)$ so $L(a', b', c', j)$

Therefore take b, c s.t. $\phi_1(a, b, c, j) \equiv \ell(a', b', c', j)$

These are unique by construction s.t. $(\phi_1 \circ \phi_2)(a, b, c, j) \equiv \ell(a'', b'', c'', j)$

Hence $\varphi_1 \circ \varphi_2$ is a homomorphism □

Corollary: If S, T, U are sets; $\varphi_1 : S \to T$, $\varphi_2 : T \to U$ are isomorphisms then $\varphi_1 \circ \varphi_2$ is also an isomorphism

Proof: Follows from the last lemma and that the composition of bijections is also a bijection □

This finally brings us to canonical definitions

Definition: Canonical definition

Given premises Γ, variables $\bar{x}$ and a formula α that's link-defined on $\bar{x}$ under Γ;

A canonical definition is any solution $f : \bar{x} \to S$ s.t. the only homomorphisms from $R^*(f\,\bar{x})$ to some $R^*(g\,\bar{x})$ s.t. g is also a solution ~~is~~ are ones that does not cause any more x_i to be aliased with another variable. In other words, $\cancel{\varphi|_{\text{range}}}$ range $g \equiv$ range f.

And $\forall g : \bar{x} \to S, \varphi : R^*(g\,\bar{x}) \to R^*(f\,\bar{x})$

In the next section, read 'fixed-point homomorphism' where 'homomorphism' is written. The definition was insufficient as it didn't account for the fixed-fixed case.

Definition: Canonical definition

Given premises Γ with free variables $\bar{x}$, a predicate P with n parameters, n variables $\bar{y}$ and a conjunction of link formula α with free variables in $\bar{x}, \bar{y}$ and $\bar{z}$; where $\bar{x}, \bar{y}$ and $\bar{z}$ are distinct lists of variables without ~~a~~ repeat.

Let $\beta = \forall \bar{y}.\, P(\bar{y}) \Leftrightarrow \exists \bar{z}.\, \cancel{\alpha}\, \overbrace{\forall w \in \bar{x}, \bar{y}, \bar{z};\, a, b, c, n.\ Reach(w, a) \Rightarrow (L(a,b,c,n) \Leftrightarrow \alpha)}^{\gamma}$

Then β link-defines P under Γ iff

Given an assignment $F : \bar{x} \to *$, there are assignments $G : \bar{y} \to *$ and $H : \bar{z} \to *$ s.t. if $F \vdash \Gamma$ then $F, G, H \vdash \overset{\gamma}{\cancel{\alpha}}$; and given any G', H' of the same type as G, H there is a homomorphism φ from the reach-graph of the images of F G H onto the reach-graph of F, G', H', that preserves variables.

If β link-defines P under Γ then the canonical-assignment $F : \bar{x}, \bar{y}, \bar{z} \to *$ is defined to be unique s.t.

- $F \vdash \Gamma$ and $F \vdash \cancel{\alpha}\,\gamma$
- Given any F' s.t. $F' \vdash \Gamma$ and $F' \vdash \alpha$ and homomorphism φ between reach-graphs

of F and F' s.t. φ preserves variables ie. $\varphi \circ F = F'$ on $\bar{x}, \bar{y}, \bar{z}$, then

$$\forall v_1^{v_2} \in \bar{x}, \bar{y}, \bar{z}.\ F'v_1 \equiv F'v_2 \Rightarrow Fv_1 \equiv Fv_2$$

Or φ is an injection on the variables.

- Given any $G : \bar{x}, \bar{y}, \bar{z} \to *$ s.t. $G \vdash \Gamma$ and $G \vdash \alpha$, There is a homomorphism φ from the reach-graphs of G to the reach-graph of F s.t. $\varphi \circ G = F$ on $\bar{x}, \bar{y}, \bar{z}$ s.t. either φ is an isomorphism or $\exists v_1, v_2 \in \bar{x}, \bar{y}, \bar{z}$. s.t. $Fv_1 = Fv_2$ and $Gv_1 \neq Gv_2$

This likely won't be very clear without a bit of discussion. Let's start with some terms. Notation like '$\bar{x}$' represents a sequence. In this instance, $\bar{x}, \bar{y}, \bar{z}$ are variable sequences. $\bar{x}$ represent background variables because they are all the free variables shared by Γ and β. Γ are the premises and β is the outer-defining-formula. $\overset{\gamma}{\wedge}\alpha$ is the

inner-defining formula. An assignment is a map from a set of variables to any node. A solution in this context is any assignment $F : \bar{x}, \bar{y}, \bar{z} \to *$ that satisfies Γ and $\overset{\gamma}{\underset{\wedge}{\cancel{\alpha}}}$. So by definition of P, F $F \vdash P\bar{y}$ and $P(\bar{y})$ holds under any assignment G that can be extended to an assignment G' that satisfies Γ and $\overset{\gamma}{\underset{\wedge}{\cancel{\alpha}}}$. Let $R^*(\bar{n})$ denote the reach-graph of all the nodes in a set $\bar{n}$ combined. Write 'reach-graph of an assignment F' to mean $R^*(\text{range}(F))$.

Next we want to create a category of assignments and homomorphisms. Let $\bar{V}$ be $\overset{\text{the}}{\text{a}}$ sequence of variables $\bar{x}, \bar{y}, \bar{z}$. Let S be the set of solutions to Γ and α. Given ~~a~~ F, G in S, let $\Phi(F, G)$ be the set of homomorphisms $\varphi_1 : R^*(\text{range}(F)) \to R^*(\text{range}(G))$ s.t. $\forall v \in \bar{V}.\ Gv = \varphi_1 Fv$ i.e. φ_1 preserves variables, paired with F and G: (φ_1, F, G). Write φ for (φ_1, F, G) and φ_1 for φ_1, $\text{dom}\,\varphi = F$, $\text{cod}\,\varphi = G$. Write $\phi' \circ \phi$ for $(\phi'_1 \circ \phi_1, F, H)$ where $\phi' \in \Phi(G, H)$ then $\phi' \circ \phi \in \Phi(F, H)$ as homo & preserve $\phi \in \Phi(F, G)$

variables: $\forall v \in \bar{V}. (\phi'_1 \circ \phi_1)(F v) = \phi'_1(\phi_1(F v))$

$= \phi'_1(G v)$ as ϕ preserves variables

$= H v$ as ϕ' preserves variables.

Let id_F denote the identity map on the reach-graph of F: (id_{F_1}, F, F). This is in $\Phi(F, F)$ as id_{F_1} can be seen to be a homomorphism and trivially preserves variables. Let $A = \bigcup_{F,G \in S} \Phi(F, G)$ be the ~~a~~ set of arrows between solutions. Then we have shown $(S, A, \circ, id, dom, cod) = \mathcal{G}$ is a <u>category</u>.

Next, we want to classify each $\varphi \in \Phi(F, G)$ as to the type of aliasing it does. There's three types of aliasing we want to check:

- Does it alias fixed-points with one another?
- Does it alias variables with fixed-points?
- Does it alias variables with other variables?

Call a a <u>fixed-point</u> of Γ and α iff for each F that solves Γ and α, $a \in R^*(range(F))$.

I've mixed up α, β, and γ quite badly here

Call a fixed-point a <u>direct</u> iff there is a variable v in $\bar{V}$ s.t. for every F that solves Γ and γ, and there is a node b s.t. either $L(\cancel{x}Fv, b, a)$ or $L(Fv, a, b)$.

I.e. a fixed-point is direct iff it is necessarily linked outwardly by some variable.

Or put simply:

$$Direct_{FP}(a, \Gamma, \gamma) \Leftrightarrow \exists v \in \bar{V}. \forall F. Solves(F, \Gamma, \gamma) \Rightarrow \exists b. L(Fv, b, a) \vee L(Fv, a, b)$$

F-F <u>aliasing</u> occurs when a fixed-point a appears in more direct links ~~that~~ than it did previously.

A homomorphism $\varphi \in \Phi(F, G)$ is <u>x-y aliasing</u> (for x-y in 'F','V' for 'fixed' and 'variable') iff there is ~~x~~ an a of type x and b of type y in F s.t. ~~$x \neq y$ and~~ $a \not\equiv b$ and $\varphi a \equiv \varphi b$. Say a is fixed type iff it is a fixed-point under some Γ and γ. Say a is variable type iff there is a variable $v \in \bar{V}$ s.t. $Fv \equiv a$.

If a map φ isn't x-y aliasing then call it <u>x-y-preserving</u>. It should be straight forward that all id_F are x-y-preserving and that ~~if F and G~~

φ and ϕ are x-y-preserving then so is $\phi \circ \varphi$.

Let C_{xy} be the subcategory of C restricted to only x-y-preserving arrows. Then it's clear that each C_{xy} and their arbitrary intersections are proper categories. Call a solution F ~~maximal iff~~ x-y-min ~~maximal~~ iff ~~F is initial in C_{xy}, and minimal iff F is terminal (where uniqueness is held up to isomorphism).~~ there is no solution G and homomorphism $\varphi \in \Phi(G, F)$ that is x-y-aliasing. And call F x-y-max ~~minimal~~ is there is no G and $\varphi \in \Phi(F, G)$ that is x-y-aliasing.

Let $C_{\overline{xy}}$ be the subcategory of C restricted to $C_{x_1 y_1} \cap C_{x_2 y_2}$ for $\{(x,y),(x_1,y_1),(x_2,y_2)\} = \{(F,F),(F,V),(V,V)\}$. That is, $C_{\overline{xy}}$ may only vary by aliasing types x and y but nothing else. Fix a solution F. Say G is x-y-minimal from F iff G is x-y-minimal and $\Phi(F,G) \neq \emptyset$. Likewise, say G is x-y-maximal from F iff G is x-y-maximal and $\Phi(G,F) \neq \emptyset$.

We want to show that, up to isomorphism, given any solution F,

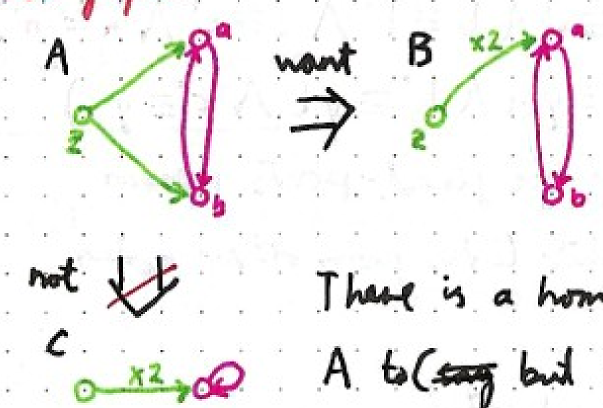
(i) There is a unique F-F maximum from F in C_{FF} *

(ii) There is a unique V-V maximum from F in $C_{\overline{VV}}$

(iii) There is a unique F-V minimum from F $C_{\overline{FV}}$

We will use these to define the canonical solution.

Proof (i): *Actually homomorphism won't cut it in this case. To alias fixed-points, we need to be moving links to isomorphic fixed-points, not compressing the fixed-point reach graph.



There is a homomorphism from A to C ~~too~~ but if A is a solution then C can't be because the fixed-points don't have an isomorphism between them. To get from A ~~to~~ to B, we want to pull the variable graph from the fixed-points till they pop off then let them re-settle and see where the links end up.

Given a link-definition $\Gamma, \overline{\nabla}, \alpha$, with fixed-point set S

A <u>fixed-point homomorphism</u> of the link definition is any map φ s.t. there is a link-map ϕ where the following all hold

- $\forall \cancel{a}, b, c, n. \cancel{a \in \mathrm{domain}(\varphi)} \wedge \cancel{a \notin S} \wedge L(\cancel{a}, b, c, n) \cancel{\Rightarrow}$
  $L(\cancel{\phi}(a, b, \cancel{c}, n))$
- $\forall a, b, c, n, a', b', c', n'. \phi(a, b, c, n) \equiv \ell(a', b', c', n') \Rightarrow$
  $(a \in S \wedge a \equiv a' \wedge b \equiv b' \wedge c \equiv c' \wedge n \equiv n')$
  $\vee (a \notin S \wedge a' \equiv \varphi a \wedge b' \equiv \varphi b \wedge c' \equiv \varphi c)$

  That is links from fixed-points remain unchanged while links from other nodes follow φ.
- φ and ϕ are situated
- ϕ is closed
- ϕ is out-link bijective
- ϕ and φ are proper maps
- φ respects pegs
- $\forall a. a \in S \Rightarrow a \simeq \varphi a$

  I.e. φ only maps fixed-points to isomorphic nodes.

This changes 'ϕ follows φ' in the homomorphic definition to 'non-fixed-points have ϕ follow φ, while fixed point links are mapped to themselves', adding that fixed points may only be mapped to isomorphic nodes (which will always also be a fixed point).

Back to the program:

Claim (i): There is a unique F-F maximum for some fixed F, in C_{FF} up to isomorphism.

Proof:

~~Take G_1, G_2 F-F maximum from F.~~

~~Take $\phi_1 \in \Phi(F, G_1)$ and $\phi_2 \in \Phi(F, G_2)$~~

~~WTS there is a $\phi_3 \in \Phi(G_1, G_2)$.~~

Let S be the set of fixed-points.

Let Π be the set of isomorphisms $\pi : S \to S$.

This is finite, as the number of permutations of a finite graph is necessarily finite.

Let T be the set ~~of~~ of subsets of S s.t.

- $\forall x, y.\ x \in S \wedge y \in S \cancel{\Rightarrow} \wedge\ y \simeq x \Rightarrow$

  $\exists! s.\ s \in T \wedge x \in s \wedge y \in s$

'Every isomorphic pair both appear in some s.'

- $\forall x.\ x \in S \Rightarrow \exists! s \in T.\ x \in s$

'Every node appears in exactly one s.'

- $\forall x, y, s.\ s \in T \wedge x \in s \wedge y \in s \Rightarrow$

  $x \in S \wedge y \in S \wedge x \simeq y$

'T contains isomorphic subsets'

- $\forall s \in T.\ \exists x \in s.$

'$\emptyset \notin T$.'

Let $\cancel{X}^{U} \subset S$ Let be a set s.t.

$\forall s \in T.\ \exists! x \in \cancel{X}^{U}_{x}.\ x \in s$

A selection from each iso set.

Let M be a one-place predicate defined as

$\forall \mu.\ M(\mu) \Leftrightarrow$

$\mathrm{Map}(\mu)$

$\wedge\ \mathrm{Proper}(\mu)$

$\wedge\ \mathrm{domain}(\mu) \equiv S$

$\wedge\ \forall x \in S.\ \mu x \simeq x$

$\wedge\ \forall x, y \in S.\ x \simeq y \Rightarrow \mu x \equiv \mu y$

I.e. μ compresses each equivalence classes to a single member.

In general say the identity-extension of a map χ to domain D is the union of χ and ι, where ι is the identity map from $D \setminus \mathrm{domain}(\chi)$ to itself. I.e.

$\forall a.\ a \in D \wedge a \notin \mathrm{domain}(\chi) \Rightarrow (a \mapsto a) \in \iota$

$\wedge\ \forall b.\ b \in \iota \Rightarrow \exists a \in D \setminus \mathrm{domain}(\chi)\ b \equiv (a \mapsto a)$

provided that $\mathrm{domain}(\chi) \subset D$.

Given any μ satisfying M, let μ' be its identity expansion to $R^* F$. Let $G = \mu' \circ F$. WTS $\mu \in \Phi(F, G) \wedge$ $\mu \in C_{FF} \wedge G$ is F-F maximum from F.