1. Contradiction derivating

In Asher [1] Nicholas Asher proposed a new quantificational type to model first-order properties. And for this type he proposed two rules for this type: ∃introduction and ∃exploitation. But these two rules together will lead to an unwanting result: subtyping hierarchies of properties would be deflated into only 2 levels, even worse, there is a contradiction. To keep the method of modeling first-order properties via existential types, we have to modify the ∃explpitation rule. I will discuss the contradiction, modification and concequences in detail.

The quantificational type is defined as

> If σ is a simple type, and τ is any expression denoting a type and x is a variable ranging over types, then $\exists \mathrm{x} \sqsubseteq \sigma\tau$ is a type. To illustrate, a term t is of this quantificational type if there is a subtype x of σ such that t is of type $\tau[\mathrm{x}]$.

Two rules of "∃" are

> ... A is any type expression with an occurrence of β and B a type expression where β does not occur, then
>
> Type theoretic∃introduction:
>
> $$\frac{\beta \sqsubseteq \alpha}{A \sqsubseteq (\exists \mathrm{x} \sqsubseteq \alpha A(\frac{\mathrm{x}}{\beta})}$$
>
> Type theoretic∃exploitation:
>
> $$\frac{\beta \sqsubseteq \alpha,\ A \sqsubseteq B}{(\exists \mathrm{x} \sqsubseteq \alpha A(\frac{\mathrm{x}}{\beta})) \sqsubseteq B}$$

From the rules and functional type we can deduce:

Theorem 1. *If we have at least 2 types, which forming a chain under subtyping relation as $a_0 \sqsubseteq ... \sqsubseteq a_i ... \sqsubseteq a_j ...$, with whenever $i \neq j$, $a_i \neq a_j$; and the functional types: $(a_0 \Rightarrow T)$, ..., $(a_i \Rightarrow T)$, ..., $(a_j \Rightarrow T)$, ..., for any type T,*

then by rules of ∃, we have: for any $n \in \mathbb{N}$, $(a_n \Rightarrow T)$ equals to $(\exists x \sqsubseteq a_{n+1}(x \Rightarrow T))$;

by transitivity of subtyping $\sqsubseteq$ we have more: for any $n, m \in \mathbb{N}$, if $n < m$, then $(a_n \Rightarrow T) = (\exists x \sqsubseteq a_{n+1}(x \Rightarrow T))$.

Corollary 2. *Given a subtype chain $a_0 \sqsubseteq ... \sqsubseteq a_i ... \sqsubseteq a_j ...$, with a_i, a_j pair-wisely distinct, for any $n, m, l \in \mathbb{N}$, if $n < l$, $m < l$, then $(a_n \Rightarrow T) = (a_m \Rightarrow T)$.*

Corollary 3. *Given a subtype chain $a_0 \sqsubseteq ... \sqsubseteq a_i ... \sqsubseteq a_j ...$, with a_i, a_j pair-wisely distinct, for any $n, m, l \in \mathbb{N}$, if $n > l$, $m > l$, then $(\exists x \sqsubseteq a_n(x \Rightarrow T)) = (\exists x \sqsubseteq a_m(x \Rightarrow T))$.*

Proof. Suppose we have a type T, and two distinctive types satisfying $a_i \sqsubseteq a_j$, thus $(a_j \Rightarrow T) \sqsubseteq (a_i \Rightarrow T)$.

From $\exists$ introduction, we have

$$1: \ (a_i \Rightarrow T) \sqsubseteq (\exists \mathrm{x} \sqsubseteq a_j(\mathrm{x} \Rightarrow T)).$$

Since $a_j \sqsubseteq a_j$, $(a_j \Rightarrow T) \sqsubseteq (a_i \Rightarrow T)$, and a_j does not occur in $(a_i \Rightarrow T)$, from $\exists$ exploitation, we have

$$2: \ (\exists \mathrm{x} \sqsubseteq a_j(\mathrm{x} \Rightarrow T)) \sqsubseteq (a_i \Rightarrow T).$$

Combining 1 and 2 we have

$$(a_n \Rightarrow T) = (\exists \mathrm{x} \sqsubseteq a_{n+1}(\mathrm{x} \Rightarrow T)).$$

The corollaries are easy to prove with the transitivity of subtying relation. □

The corollaries lead to a contradiction, since we have assumed that a_i, a_j are distinct, thus $(a_j \Rightarrow T)$ and $(a_i \Rightarrow T)$ would never the same.

This contradition comes from the quantificational type. One can image this type $(\exists \mathrm{x} \sqsubseteq \alpha A(\frac{\mathrm{x}}{\beta}))$ as an expansion from the original type A. For two types A and B with $A \sqsubseteq B$, since it's possible that some of $A(\frac{\mathrm{x}}{\beta})$s are larger than B, so this expansion would make A larger than B, so the contradiction occurs. Hence to avoid the contradiction we have to modify the $\exists$exploitation rule, to add some constrained conditions .

A quantificational type is used to model a first-order property. The most natural idea is that first-order properties are of type "$(\mathrm{x} \Rightarrow T)$", for x a subtype of e, and T the type of proposition. But types like $(\mathrm{x} \Rightarrow T)$ are monotonic reversely for x. "$\Rightarrow$" reverses the antecedents' order. For any type c, d, c$\sqsubseteq$d iff $(\mathrm{d} \Rightarrow T) \sqsubseteq (\mathrm{c} \Rightarrow T)$, while we prefer to remain the order, for example, maintaining the order from "tiger is a subtype of animal" to "being a tiger is a subtype of being an animal". This is the motivation that the quantificational type is introduced.

By definition $\exists \mathrm{x} \sqsubseteq \sigma\tau(\mathrm{x})$ equals to $\bigvee_{\mathrm{x} \sqsubseteq \sigma} \tau(\mathrm{x})$, given β is a simple type symbol occuring free in τ, and x replaces every free occurence of β. So any type of first-order properties, $\exists \mathrm{x} \sqsubseteq \sigma(\mathrm{x} \Rightarrow T)$, would be in fact a disjunctive type (finite or infinite depends on the subtypes' cardinal of σ). For example, since we have physical, informational objects's types, p, i, are both subtypes of entities', e; and cats' type, c, is a subtype of physical objects' hence also a subtype of e; etc.; then $\exists \mathrm{x} \sqsubseteq \mathrm{e}(\mathrm{x} \Rightarrow T)$ (being an entity) would be

$$(\mathrm{e} \Rightarrow T) \vee (\mathrm{p} \Rightarrow T) \vee (\mathrm{i} \Rightarrow T) \vee (\mathrm{c} \Rightarrow T) \vee \ldots .$$

And $\exists \mathrm{x} \sqsubseteq \mathrm{p}(\mathrm{x} \Rightarrow T)$ "being an physical object" would be

$$(\mathrm{p} \Rightarrow T) \vee (\mathrm{c} \Rightarrow T) \vee \ldots .$$

Since a type always has more subtypes than any of its subtypes, the funtional types derived from that type would be more than any of its subtypes, hence the order is preserved. But from this analysis it's also easy to see the source of the contradiction, i.e., some of the disjuncts will be too large as the subtyping goes on.

2. Modification on ∃ exploitation rule; Branching

We have analyzed the nature of existential types. Now it's natural to require all the disjunct are small enough to keep safe. We require on the $A(\frac{\gamma}{\beta})$s such that every $A(\frac{\gamma}{\beta})$ is a subtype of B. Thus our new ∃ exploitation rule would be

Given A is any type expression with an occurrence of β and B a type expression where β does not occur, then

Type theoretic$\exists^*$exploitation:

$$\frac{\beta \sqsubseteq \alpha,\ A \sqsubseteq B,\ A(\frac{\gamma}{\beta}) \sqsubseteq B(\text{for any subtype } \gamma \text{ of } \alpha)}{(\exists \mathrm{x} \sqsubseteq \alpha A(\frac{\mathrm{x}}{\beta})) \sqsubseteq B}.$$

$\exists^*$exploitation is sound. Since every discjunt of $(\exists \mathrm{x} \sqsubseteq \alpha A(\frac{\mathrm{x}}{\beta}))$ is a subtype of B now, and subtyping relation is closed under disjuntion operator, hence trivially $(\exists \mathrm{x} \sqsubseteq \alpha A(\frac{\mathrm{x}}{\beta})) \sqsubseteq B$.

Our previous proof of theorem 1 will be stuck, so the contradiction is prevented. Replacing the rule $\exists^*$exploitation for ∃exploitation, since there is some subtype a_k of a_j such that $(a_i \Rightarrow T) \sqsubseteq (a_k \Rightarrow T)$, which not satisfying our restriction "$A(\frac{\gamma}{\beta}) \sqsubseteq B(\text{for any subtype } \gamma \text{ of } \alpha)$".

3. Consequence of the modified rule $\exists^*$exploitation

Further, with this new exploitation rule and the definition of existential types, i.e. the representation of first-order properties, we have an interesting result: the structure of first-order properties (under subtyping relation) is forking (at least binary branched). It means that every first-order property would have (if it has a proper sub-property) at least 2 other properties as its proper subtypes.

To proof this, let's first check two lemmas.

Lemma 4. *For any subtyping chain of* e *which has a minimium element* a*, the chain of corresponding* $(x \Rightarrow T)$ *has a maximium element namely* $(a \Rightarrow T)$*, hence for any* b *in this chain,*$\exists \mathrm{xx} \sqsubseteq \mathrm{b}(\mathrm{x} \Rightarrow T) = \exists \mathrm{xx} \sqsubseteq \mathrm{a}(\mathrm{x} \Rightarrow T)$.

Proof. Since a is the minimium element, having no other subtypes but itself, by definition of existential types we have $\exists xx \sqsubseteq a(a \Rightarrow T) = (\mathrm{a} \Rightarrow T)$.

Since for every b in this chain we have a⊑b, thus $(\mathrm{b} \Rightarrow T) \sqsubseteq (\mathrm{a} \Rightarrow T)$; and for any subtype c of b, $(\mathrm{c} \Rightarrow T) \sqsubseteq (\mathrm{a} \Rightarrow T)$; b doesn't occur in $(\mathrm{a} \Rightarrow T)$; by $\exists^*$exploitation we have $\exists xx \sqsubseteq b(x \Rightarrow T) \sqsubseteq (\mathrm{a} \Rightarrow T)$.

The other direction $(\mathrm{a} \Rightarrow T) \sqsubseteq \exists xx \sqsubseteq b(x \Rightarrow T)$ is trivial.

Hence we have $\exists xx \sqsubseteq b(x \Rightarrow T) = \exists xx \sqsubseteq a(x \Rightarrow T)$. □

Lemma 5. *If a set/class* A *of several types could form a tree under subtyping relation, with the root* r *is the greatest element,* B *is the set of all leaves(the minimium elements) in* A*, then*

$$\exists \mathrm{x} \mathrm{x} \sqsubseteq \mathrm{r}(\mathrm{x} \Rightarrow T) = \bigvee_{\mathrm{b} \in \mathrm{B}} (\mathrm{b} \Rightarrow T).$$

Proof. By lemma 7 and the definition of existential types. □

It should be noted that by lemma 5, in order to make a first-order property meaningful, we should demand urelements in our types, and $\perp$ is not a subtype of any basic type. Or else the first-order property is either unintelligible or absurd. And since on the bottom level types are incompatible each other, the type theory is akin *La Monadologie.*

Theorem 6. *The type structure of first-order properties is forking* (*at least binary branched*)*.*

Proof. By lemma 7&8, since every un-forking branch would crash into mere one first-order property. □

Theorem 9 shows that for any two property, if one is the unique imediate sub-property of the other, then they are the same property. For example, if "being extensive" is the unique imediate sub-property of "being physical", then "being extensive" is the same first-order property as "being physical".

4. Indiscernibility

We may consider the transformation from a to (a $\Rightarrow T$), and to $\exists xx \sqsubseteq a(x \Rightarrow T)$ as maps, mapping some type a to a-like first-order property $\exists xx \sqsubseteq a(x \Rightarrow T)$: given any basic type a, the corresponding first-order property is $\exists xx \sqsubseteq a(x \Rightarrow T)$, for short, fop(a). In our theory the map fop have all the basic types as its domain, and the first-order property part of quantificational types as its codomain. It's easy to see that fop is a homomorphism under subtyping relation.

In a chain of basic types, even if the nodes are distinct, from lemma 7 we still have their corresponding first-order-property types are indisernibles. This kind of reprentation for first-order properties reveals the meaning of "identity of indicernibles" principle: types which make no privileged differences are not real types. Given any two basic types a, b, if a has only b as its subtype, then it's meanningless to distinguish b from a.

It's reasonable for the nature of typing "to distinguish". A type sysytem, as a way of sorting entities, is a classification of entities. If a type (not including the toppest type, ENTITY) makes no contribution to the classification, then this type is a redundance. By Occam's Razor we should shave it. Using quantificational types we deflate these redundances into one type, which persists the distinguishing nature of typing.

Some may argues that some type indeed has another type as its unique proper subtype, for example, EXTENSIVE$\sqsubseteq$PHYSICAL. I agree with this subtyping, but these two basic types forms a undividable cluster, since everything that is extensive is physical, vice visa. The types EXTENSIVE and PHYSICAL merge into one cluster. In fact this subtyping is not complete: we have PHYSICAL$\sqsubseteq$EXTENSIVE. Now by antisymmetry of $\sqsubseteq$, PHYSICAL=EXTENSIVE. Some may insists that at most we could admit they are equal, but never the same. He may argue that equal properties are still differents, so the types should also be different. Even every triangle is a trilateral, the property "being a triangle" is still not the same as "being a trilateral". Though the former is a property on angle, while the latter is on lateral.

5. Getting rid of indiscernibility

The indicernibility result comes from two sources: 1, the representaion of first-order properties in quantificational types; 2, the antisymmetry of subtyping relation $\sqsubseteq$. In order to get rid of indisernibility, one may abondon any of these sources. I have tried to defend the reason of quantification types, and I will consider the consequence of dropping antisymmetry.

Antisymmetry of $\sqsubseteq$ says, for any types a, b, if a$\sqsubseteq$b and b$\sqsubseteq$a, then a=b. There is no two distinct types each being a subtype of the other. By dropping antisymmetry, our lemmas will fail hence we have chains and an unforking type theory.

But there are payoffs that we may encounter deductive (still have equivallent) and philosophical difficulties. On one hand, any deduction relying on antisymmetry would be invalid unless we find a alternative without antisymmetry. And would a type theory without antisymmetry fruitful enough? On the other hand, we may encounter redundences and deletion. Redundences come from ontology explosion, since now we don't have an standard to count two types as one, then whether a$\sqcap$a equals to a, or a$\sqcap$a$\sqcap$a, ..., etc? There will be too many distinct types. Deletion comes from Quine's standpoint. Since antisymmetry is a standard of identity, which telling that under what condition two types are one, according to his famous remark "no entity without identity", cancelling antisymmetry would lead to the deletion of some types.

6. Modified type composition logic

Since we have modified the $\exists$-exploitation rule to avoid contradition, now we can define our new type composition logic.

Definition 7. Alphabet. At first, we specify a language $\mathscr{L}$, whose alphabet is,

an at most countable list x_0, x_1, x_2, ... for basic types variables and an at most countable list c_0, c_1, c_2, ... for basic types constants;

type constructors $\sqcup$, $\sqcap$, $\Rightarrow$, $\cdot$,$\exists$;

subtyping relation $\sqsubseteq$;

connectives $\vee$, $\wedge$, $\rightarrow$;

identity and equivalence symbols $=$, $\equiv$;
coercion $\rhd$, $\lhd$;
brackets), (;
particularly we use $\perp$ for a special basic type, the bottom type.

Definition 8. Terms. There is only three kinds of terms,
(1) a basic type symbol is a term;
(2) for any terms t, s, any $\circledast$ in $\{\sqcup, \sqcap, \Rightarrow, \cdot\}$, $(t \circledast s)$ is a term;
(3) for any term t, if a constant c is contained in t, then for any variable x and any constant d,$\exists \mathrm{x} \sqsubseteq \mathrm{d}t(\frac{\mathrm{x}}{\mathrm{c}})$ is a term, where $t(\frac{\mathrm{x}}{\mathrm{c}})$ stands for substituting one or more occurence c with x in t.

Definition 9. Formulae. There is only one kind of formulae.
For any terms t, s, any $\circledast$ in $\{\sqsubseteq, \vee, \wedge, \rightarrow, =, \equiv, \rhd, \lhd\}$, $(t \circledast s)$ is a formula.
Usually the brackets are omitted when no ambiguity occurs.

Definition 10. Type System. Corresonding to terms now we define a type system $\mathscr{T}$, $\mathscr{T}$ is a tuple $< T, \sqsubseteq >$ where
$\sqsubseteq$ is the subtyping relation (binary, reflexive, transitive) on T;
T includes a collection of basic types;
T is the minimium collection that for any types α, β in T, any type expression A denoting α, the union type $\alpha \sqcup \beta$ (where there is a γ, $\alpha \sqsubseteq \gamma$, $\beta \sqsubseteq \gamma$), intersection type $\alpha \sqcap \beta$, function type $\alpha \Rightarrow \beta$, dot type$\alpha \cdot \beta$, and existential type $\exists \mathrm{x} \sqsubseteq \mathrm{d}A(\frac{\mathrm{x}}{\mathrm{c}})$ are also in T.

The relationship of types obeys serveral rules. Given any types α, β, γ, δ, and any set of type formulae Γ,

comutative:

$$\frac{\Gamma}{(\alpha \circledast \beta) \sqsubseteq (\beta \circledast \alpha)},$$

for any $\circledast$ in $\{\sqcup, \sqcap, \cdot, =, \equiv\}$;

substitution:

$$\frac{\Gamma,\ (\alpha \circledast \beta)}{A \sqsubseteq A(\frac{\beta}{\alpha})},$$

for any $\circledast$ in $\{=, \equiv\}$;

coercive-substitution:

$$\frac{\Gamma,\ (\alpha \circledast \beta)}{A \circledast A(\frac{\beta}{\alpha})},$$

for any $\circledast$ in $\{\rhd, \lhd\}$;

$\sqcup_+$:

$$\frac{\Gamma}{\alpha \sqsubseteq (\alpha \sqcup \beta)},$$

provided there is a δ with $\alpha \sqsubseteq \delta$ and $\beta \sqsubseteq \delta$;

$\sqcup_-$: if $\frac{\Gamma}{\alpha\sqsubseteq\gamma}$, and $\frac{\Gamma}{\beta\sqsubseteq\gamma}$, then

$$\frac{\Gamma}{(\alpha\sqcup\beta)\sqsubseteq\gamma};$$

$\sqcap_+$:

$$\frac{\Gamma}{(\alpha\sqcap\beta)\sqsubseteq(\alpha\sqcap\beta)};$$

$\sqcap_-$:

$$\frac{\Gamma}{(\alpha\sqcap\beta)\sqsubseteq\alpha};$$

$\Rightarrow$:

$$\frac{\Gamma,\ \alpha\sqsubseteq\beta,\ \gamma\sqsubseteq\delta}{(\beta\Rightarrow\gamma)\sqsubseteq(\alpha\sqsubseteq\delta)};$$

$\cdot_+$:

$$\frac{\Gamma,\ \alpha\sqcap\beta{=}\bot}{\alpha\rhd(\alpha\cdot\beta)},$$

provided there is a δ with $\alpha\sqsubseteq\delta$ and $\beta\sqsubseteq\delta$, and $\alpha\rhd(\alpha\cdot\beta)$ means type α is coerced to $\alpha\cdot\beta$;

$\cdot_-$:

$$\frac{\Gamma}{A\lhd A(\frac{\alpha}{\alpha\cdot\beta})},$$

where A is a type expression, $A(\frac{\alpha}{\alpha\cdot\beta})$ is the result by replacing one or more occurence of $\alpha\cdot\beta$ with α;

$\exists$introduction:

$$\frac{\Gamma,\ \beta\sqsubseteq\alpha}{A\sqsubseteq(\exists\mathrm{x}\sqsubseteq\alpha A(\frac{\mathrm{x}}{\beta})};$$

$\exists^*_-$ exploition:

$$\frac{\Gamma,\ \beta\sqsubseteq\alpha,\ A\sqsubseteq B}{(\exists\mathrm{x}\sqsubseteq\alpha A(\frac{\mathrm{x}}{\beta}))\sqsubseteq B},$$

provided $A(\frac{\gamma}{\beta})\sqsubseteq B$(for any subtype γ of α).

As for rules of these form,

$$\frac{\Gamma,\alpha}{\beta},$$

it would be easier to understand Γ as a set/collection of types, and we have a set Δ(not necessarilly different from Γ) satisfying $\alpha \in \Delta$. So this rule says if $\alpha \in \Delta$, after we make a union of $\Gamma \cup \Delta$, we can infer $\beta \in \Gamma \cup \Delta$.

from $\Gamma, \alpha \vdash \beta \Rightarrow \alpha$

confusion on the levels. language, syntactic, semantic.

What's a logic system? Montague used an artificial language, intensional logic, as a medium.

modus ponen

the uniqueness of pullback

.

[1] Asher, N.: 2011, *Lexical Meaning in Context*, Cambridge University Press.