# TAS
# Cours 05 - Typage Objet

Romain Demangeon

TAS - M2 STL

17/10/2024

SORBONNE UNIVERSITÉ

- Formules logiques avec l'implication.

$$\phi ::= A \mid \phi \Rightarrow \phi$$

- $\Gamma$ : ensemble d'hypothèses (de formules),
- Jugements $\Gamma \vdash \phi$ : "$\phi$ est prouvable avec les hypothèses $\Gamma$"
- Séquents Intuitionnistes :

$$\textbf{(Ax)}\,\frac{}{\Gamma, A \vdash A} \qquad \textbf{(MP)}\,\frac{\Gamma \vdash A \Rightarrow B \qquad \Gamma \vdash A}{\Gamma \vdash B} \qquad \textbf{(I)}\,\frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}$$

- Formules prouvables :

$$\textbf{(I)}\,\dfrac{\textbf{(I)}\,\dfrac{\textbf{(I)}\,\dfrac{\textbf{(MP)}\,\dfrac{\textbf{(MP)}\,\dfrac{\textbf{(Ax)}\,\dfrac{}{\Gamma \vdash A \Rightarrow (B \Rightarrow C)} \qquad \textbf{(Ax)}\,\dfrac{}{\Gamma \vdash A}}{\Gamma \vdash B \Rightarrow C} \qquad \textbf{(MP)}\,\dfrac{\textbf{(Ax)}\,\dfrac{}{\Gamma \vdash A \Rightarrow B} \qquad \textbf{(Ax)}\,\dfrac{}{\Gamma \vdash A}}{\Gamma \vdash B}}{A \Rightarrow B \Rightarrow C, A \Rightarrow B, A \vdash C}}{A \Rightarrow B \Rightarrow C, A \Rightarrow B \vdash A \Rightarrow C}}{A \Rightarrow B \Rightarrow C \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow C}}{\vdash (A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C}$$

# Curry-Howard : Correspondance

- Formules de SI $\leftrightarrow$ Types de $\lambda_{ST}$
- Preuves de SI $\leftrightarrow$ Dérivation de typage de $\lambda_{ST}$
  - $=$ Termes de $\lambda_{ST}$
  - car le système de types est dirigé par la syntaxe

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : A \to (B \to C)}\,(\textbf{Var}) \qquad \dfrac{}{\Gamma \vdash z : A}\,(\textbf{Var})}{\Gamma \vdash x\ z : B \to C}\,(\textbf{App}) \qquad \dfrac{\dfrac{}{\Gamma \vdash y : A \to B}\,(\textbf{Var}) \qquad \dfrac{}{\Gamma \vdash z : A}\,(\textbf{Var})}{\Gamma \vdash y\ z : B}\,(\textbf{App})}{x : A \to B \to C, y : A \to B, z : A \vdash \lambda(x\ z)\ (y\ z) : C}\,(\textbf{App})}{x : A \to B \to C, y : A \to B \vdash \lambda z.(x\ z)\ (y\ z) : A \to C}\,(\textbf{Abs})}{x : A \to B \to C \vdash \lambda yz.(x\ z)\ (y\ z) : (A \to B) \to A \to C}\,(\textbf{Abs})}{\vdash S : (A \to B \to C) \to (A \to B) \to A \Rightarrow C}\,(\textbf{Abs})
$$

- ??? de SI $\leftrightarrow$ Réduction de $\lambda_{ST}$

# Curry-Howard : Elimination des coupures

- Opération de transformation des preuves.
- Utilisation d'un lemme :

$$(\mathbf{App})\dfrac{(\mathbf{I})\dfrac{\dfrac{\mathcal{P}_1}{\Gamma, A \vdash B}}{\Gamma \vdash A \Rightarrow B} \qquad \dfrac{\mathcal{P}_2}{\Gamma \vdash A}}{\Gamma \vdash B} \qquad \longrightarrow \qquad \dfrac{\mathcal{P}_1[\mathcal{P}_2]}{\Gamma \vdash B}$$

avec $\mathcal{P}_1[\mathcal{P}_2]$ la preuve obtenue en prenant $\mathcal{P}_1$ et en remplaçant tous les $(\mathbf{Ax})\dfrac{}{\Gamma', A \vdash A}$ par $\dfrac{\mathcal{P}_2}{\Gamma' \vdash A}$

- on simplifie une preuve en *inlinant* un lemme à tous les endroits où on en avait besoin.
- l'enchainement de $(\mathbf{App})$ avec $(\mathbf{I})$ à gauche s'appelle une coupure, le processus correspondant à la réduction est l'élimination des coupures.
- éliminer une coupure peut faire grossir la preuve et ajouter des coupures.
  - en copiant plusieurs fois les coupures dans $\mathcal{P}_\in$
- l'élimination des coupures termine.

# Curry-Howard : Logique Classique

- Calcul des Séquents classique :

$$(\textbf{TE})\frac{}{\vdash A \vee A} \quad \textbf{ou} \quad (\textbf{Abs})\frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A} \quad \textbf{ou}$$

$$(\textbf{PL})\frac{\Gamma \vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A}{\Gamma \vdash A}$$

- comment introduire $A \vee B$ dans $\lambda_{ST}$ ?

# Curry-Howard : Logique Classique

- Calcul des Séquents classique :

$$(\textbf{TE})\frac{}{\vdash A \vee A} \qquad \textbf{ou} \qquad (\textbf{Abs})\frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A} \qquad \textbf{ou}$$

$$(\textbf{PL})\frac{\Gamma \vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A}{\Gamma \vdash A}$$

- comment introduire $A \vee B$ dans $\lambda_{ST}$ ?

$$(\texttt{SumG})\frac{\Gamma \vdash M : T}{\Gamma \vdash \texttt{g} : M : T \vee U} \qquad (\texttt{SumG})\frac{\Gamma \vdash M : U}{\Gamma \vdash \texttt{d} : M : T \vee U}$$

$$(\texttt{Sw})\frac{\Gamma \vdash M : T \vee U \qquad \Gamma, x : T \vdash N_1 : S \qquad \Gamma, x : U \vdash N_2 : S}{\Gamma \vdash \texttt{sw d} : M : N_1 + N_2 : S}$$

soit

$$(\textbf{LI})\frac{\Gamma \vdash T}{\Gamma \vdash T \vee U} \qquad (\textbf{RI})\frac{\Gamma \vdash U}{\Gamma \vdash T \vee U} \qquad (\textbf{E})\frac{\Gamma \vdash T \vee U \qquad \Gamma, T \vdash S \qquad \Gamma, U \vdash S}{\Gamma \vdash S}$$

# Curry-Howard : Logique Classique (II)

- il faut complexifier $\lambda$ pour obtenir des calculs en relation avec $SK$
- le $\lambda\mu$-calcul permet de définir des termes nommés:

$$M ::= x \mid \lambda x.M \mid M\ M \mid \mu\alpha.E \qquad E ::= [\alpha]\ M$$

- la réduction structurelle permet, dans un terme liant le nom $\alpha$, de distribuer un argument $N$ à tous les sous-termes nommés par $\alpha$.

$$\frac{}{(\mu\alpha.M)\ N \longrightarrow \mu\alpha.M[[\alpha](N\ M')/[\alpha]M']}$$

- le système de types standard de $\lambda\mu$ correspond à la déduction naturelle classique.
- $\overline{\lambda}\mu\tilde{\mu}$ correspond aux séquents classiques

$$C ::= [V|E] \qquad V ::= x \mid \lambda x.V \mid e\ v \mid \mu\alpha.c \qquad E ::= \alpha \mid \alpha\lambda.e \mid v\ e \mid \overline{\mu}x.c$$

- avec la réduction

$$\frac{}{[\lambda x.V|V'\ E] \longrightarrow [V'|\overline{\mu}x.[V|E]]} \qquad \frac{}{[E'\ V|\alpha\lambda.E] \longrightarrow [\mu\alpha.[V|E]|E'}$$

$$\frac{}{\mu\alpha.[V|\alpha]} \longrightarrow V \qquad \frac{}{[\mu\alpha.C|E] \longrightarrow C[E/\alpha]} \qquad \frac{}{[V|\overline{\mu}.C] \longrightarrow C[V/x]}$$

$$\frac{}{\overline{\mu}x.[x|E]} \longrightarrow E$$

- termes, contextes, et commandes manipulent le flot de contrôle.

# Cube de Barendregt

- trois directions pour enrichir $\lambda_{ST}$
- termes dépendants de types : Polymorphisme
  - Système F présenté avec instantiation explicite et réduction typée

$$M ::= x \mid \lambda x.M \mid M\ M \mid \Lambda T.M \mid T \qquad T ::= \alpha \mid T \to T \mid \forall \alpha.T$$

$$\frac{\Gamma \vdash \Lambda X.M : \forall \alpha.T}{\Gamma \vdash \Lambda X.M\ U \longrightarrow \Gamma \vdash M : T[U/X]}$$

  - on peut définir $\delta$ comme $\Lambda Y.\lambda x.(x\ Y \to Y)\ (x\ Y)$
  - et $I$ comme $\Lambda X.\lambda x.x$
  - et typer $\delta\ (\forall \alpha.(\alpha \to \alpha))\ I$.
- types dépendant de termes (appelés "types dépendants")
  - formellement $T ::= \alpha \mid T \to T \mid \pi x.T$
  - permet de définir, par exemple, 4 `vect`, les vecteurs de taille inférieure à 4.
- types dépendant de types (constructeurs de types)
  - formellement $T ::= \alpha \mid T \to T \mid \Pi X.T$
  - permet de définir `Liste A`, les listes d'éléments de types $A$,
  - hierarchie de sortes (comme en PAF)

# Curry-Howard : Correspondances

- Le Calcul des Constructions *ferme* le cube de Barendregt
- Quelques correspondances de Curry-Howard connues :
  - SI $\leftrightarrow$ $\lambda_{ST}$
  - SK $\leftrightarrow$ $\overline{\lambda}\mu\tilde{\mu}$
  - Arithmétique de Peano $\leftrightarrow$ Système F
  - Logique de Hilbert $\leftrightarrow$ Logique Combinatoire
  - LI d'ordre supérieur $\leftrightarrow$ Calcul des Constructions.
  - $\lambda$ linéaires $\leftrightarrow$ Logiques Linéaires.

# Langages objets

- Idée : rapprocher les données de leurs traitements.
- années 80 :
  - monde de la recherche : langages et IA.
  - *Simula*, *SmallTalk*
- années 90 :
  - monde de l'industrie.
  - *C++*, *CLOS*, *Delphi*, *Java*, *C#*, *Python*, *Javascript*, *Ruby*, ...
  - génie logiciel (diagrammes)
- années 2000 :
  - développement de méthodes et d'outils
  - généricité, tests, multi-paradigme, modèles.

# Langages Objets : Typage

- dynamique nominal (*SmallTalk*)
  - à l'exécution : la variable $x$ contient-elle un objet de type/classe $C$ ?
- dynamique canardesque (*Python*)
  - à l'exécution : la variable $x$ contient-elle un objet avec une méthode $m$ ?
- statique nominal (*Java*)
  - à la compilation : la variable $x$ contient-elle un objet de classe $C$ ?
- statique structurel (*OCaml*)
  - à la compilation : la variable $x$ contient-elle un objet avec une méthode $m$ ?
- à prototype (*OCaml*)
  - à l'exécution : la variable $x$ contient-elle un objet dont la classe cachée (obtenue depuis un prototype et des mises-à-jour) contient-elle une méthode $m$ ?

# Objets dans $\lambda$

- On dispose d'un ensemble infini d'étiquettes $\mathtt{m}$
- On ajoute à la syntaxe la construction explicite d'un objet et un appel d'attribut/méthode :

$$M ::= \ldots \mid \begin{bmatrix} \mathtt{m}_1 & : & M, \\ \mathtt{m}_2 & : & M, \\ \ldots & & \\ \mathtt{m}_n & : & M \end{bmatrix} \mid M.\mathtt{m}$$

- Sémantique

$$(\mathtt{obj})\frac{M \longrightarrow M'}{M.\mathtt{m} \longrightarrow M'.\mathtt{m}} \qquad (\mathtt{call})\frac{}{[\ldots, \mathtt{m} : M, \ldots].\mathtt{m} \longrightarrow M}$$

- Un objet est un $n$-uplet avec des étiquettes.
  - un type produit.
- Le point . est un projecteur par étiquette.

# Objets dans $\lambda$ : Typage

- Types :

$$T ::= \ldots \mid \left\{ \begin{array}{lcl} \mathtt{m}_1 & : & T, \\ \mathtt{m}_2 & : & T, \\ \ldots & & \\ \mathtt{m}_n & : & T \end{array} \right\}$$

- Typage :

$$(\mathtt{Obj})\frac{\Gamma \vdash M_1 : T_1 \qquad \ldots \qquad \Gamma \vdash M_n : T_n}{\Gamma \vdash [\mathtt{m}_1 : M_1, \ldots, \mathtt{m}_n : M_n] : \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_n : T_n\}}$$

$$(\mathtt{Call})\frac{\Gamma \vdash M : \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_i : T_i, \ldots, \mathtt{m}_n : T_n\}}{\Gamma \vdash M.\mathtt{m}_i : T_i}$$

# Objets dans $\lambda$ : Ordre des attributs

- On force les étiquettes d'un même [] (dans les termes) ou d'un même {} (dans les types) à être différente.
- Contrairement aux $n$-uplets, l'ordre n'a pas d'importance.
- Congruence structurelle :
  - si $\sigma$ permutation de $[\![1; n]\!]$ :

$$\begin{bmatrix} \mathtt{m}_1 & : & M_1, \\ \mathtt{m}_2 & : & M_2, \\ \dots & & \\ \mathtt{m}_n & : & M_n \end{bmatrix} \equiv \begin{bmatrix} \mathtt{m}_{\sigma(1)} & : & M_{\sigma(1)}, \\ \mathtt{m}_{\sigma(2)} & : & M_{\sigma(2)}, \\ \dots & & \\ \mathtt{m}_\sigma(n) & : & M_{\sigma(n)} \end{bmatrix}$$

$$\begin{Bmatrix} \mathtt{m}_1 & : & T_1, \\ \mathtt{m}_2 & : & T_2, \\ \dots & & \\ \mathtt{m}_n & : & T_n \end{Bmatrix} \equiv \begin{Bmatrix} \mathtt{m}_{\sigma(1)} & : & T_{\sigma(1)}, \\ \mathtt{m}_{\sigma(2)} & : & T_{\sigma(2)}, \\ \dots & & \\ \mathtt{m}_\sigma(n) & : & T_{\sigma(n)} \end{Bmatrix}$$

  - $\equiv$ descend récursivement à l'intérieur des termes et des types (congruence). Par exemple :

$$\frac{M \equiv M'}{\lambda x.M \equiv \lambda x.M'} \qquad \frac{T_1 \equiv T_1' \qquad \dots T_n \equiv T_n'}{\{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_n : T_n\} \equiv \{\mathtt{m}_1 : T_1', \dots, \mathtt{m}_n : T_n'\}}$$

SORBONNE UNIVERSITÉ

# Objets dans $\lambda$ : Exemple

- deux points du plan :

$$O = \begin{bmatrix} \mathtt{a} & : & 0 \\ \mathtt{o} & : & 0 \end{bmatrix} \qquad P_1 = \begin{bmatrix} \mathtt{a} & : & 2 \\ \mathtt{o} & : & 2 \end{bmatrix}$$

$$\mathtt{let}\ dist = \lambda p_1, p_2.\mathtt{rc}\ ((p_1.\mathtt{a} - p_2.\mathtt{a})^2 + (p_1.\mathtt{o} - p_2.\mathtt{o})^2)\ \mathtt{in}\ M$$

  avec un type **F** pour les constantes à virgule flottante et ses primitives ($\mathtt{rc}$ = "racine carré").
- des autres points :

$$P_2 = \begin{bmatrix} \mathtt{a} & : & 2 \\ \mathtt{o} & : & 2 \\ \mathtt{p} & : & 2 \end{bmatrix} \qquad P_3 = \begin{bmatrix} \mathtt{a} & : & 1 \\ \mathtt{o} & : & -1 \\ \mathtt{c} & : & \text{"bleu"} \end{bmatrix}$$

- typage de *dist* :

$$\dfrac{\dfrac{(\mathtt{Call})\dfrac{(\mathtt{Var})\dfrac{}{\Gamma \vdash p_1 : \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}\}}}{\Gamma \vdash p_1.\mathtt{a} : \mathbf{F}}}{\cdots} \quad \cdots \quad \dfrac{(\mathtt{Call})\dfrac{(\mathtt{Var})\dfrac{}{\Gamma \vdash p_2 : \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}\}}}{\Gamma \vdash p_2.\mathtt{o} : \mathbf{F}}}{\cdots}}{\dfrac{p_1 : \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}\}, p_2 : \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}\} \vdash \mathtt{rc}\ ((p_1.\mathtt{a} - p_2.\mathtt{a})^2 + (p_1.\mathtt{o} - p_2.\mathtt{o})^2) : \mathbf{F}}{\vdash dist : \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}\} \to \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}\} \to \mathbf{F}}}$$

- critiques :
  - on peut avoir *dist* $O$ $P_1$
  - on ne peut pas avoir *dist* $O$ $P_2$
  - on voudrait que *dist* soit une méthode des points.

# Objets dans $\lambda$ : soi-même

- points avec méthode `dist` :

$$\left\{\begin{array}{lll} \mathtt{a} & : & \mathbf{F}, \\ \mathtt{o} & : & \mathbf{F} \\ \mathtt{dist} & : & \left\{\begin{array}{lll} \mathtt{a} & : & \mathbf{F}, \\ \mathtt{o} & : & \mathbf{F} \end{array}\right\} \to \mathbf{F} \end{array}\right\}$$

# Objets dans $\lambda$ : soi-même

- points avec méthode `dist` :

$$\left\{ \begin{array}{lcl} \texttt{a} & : & \mathbf{F}, \\ \texttt{o} & : & \mathbf{F} \\ \texttt{dist} & : & \left\{ \begin{array}{lcl} \texttt{a} & : & \mathbf{F}, \\ \texttt{o} & : & \mathbf{F} \end{array} \right\} \to \mathbf{F} \end{array} \right\}$$

- il manque un moyen de parler de soi

$$M ::= \dots \mid \heartsuit$$

$$O' = \left[ \begin{array}{lcl} \texttt{a} & : & 0 \\ \texttt{o} & : & 0 \\ \texttt{dist} & : & \lambda p.\texttt{rc}\ ((\heartsuit.\texttt{a} - p.\texttt{a})^2 + (\heartsuit.\texttt{o} - p.\texttt{o})^2) \end{array} \right]$$

- il faut déplier dans la sémantique :

$$(\texttt{call})\ \frac{}{[\dots, \texttt{m} : M, \dots].\texttt{m} \longrightarrow M[\{[\dots, \texttt{m} : M, \dots]/\heartsuit\}]}$$

  - quand on ouvre un objet pour prendre un de ses attributs, on remplace le $\heartsuit$ par l'objet lui-même.
  - Exemple :

$$O'.\texttt{dist}\ P_1 \longrightarrow (\lambda p.\texttt{rc}\ ((O'.\texttt{a} - p.\texttt{a})^2 + (O'.\texttt{o} - p.\texttt{o})^2))\ P_1$$

$$\longrightarrow \texttt{rc}\ ((O'.\texttt{a} - P_1.\texttt{a})^2 + (O'.\texttt{o} - P_1.\texttt{o})^2) \longrightarrow\longrightarrow \dots$$

# Soi-même : Typage

- Il faut modifier la règle de typage :

$$(\texttt{Obj})\ \frac{\Gamma, \heartsuit : \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\} \vdash M_1 : T_1 \qquad \ldots \qquad \Gamma, \heartsuit : \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\} \vdash M_n : T_n}{\Gamma \vdash [\texttt{m}_1 : M_1, \ldots, \texttt{m}_n : M_n] : \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\}}$$

- exemple, on rappelle $O' = \left[\begin{array}{lcl} \texttt{a} & : & 0 \\ \texttt{o} & : & 0 \\ \texttt{dist} & : & \lambda p.\texttt{rc}\ ((\heartsuit.\texttt{a} - p.\texttt{a})^2 + (\heartsuit.\texttt{o} - p.\texttt{o})^2) \end{array}\right]$,

  on pose $\mathcal{P} = \left\{\begin{array}{lcl} \texttt{a} & : & \mathbf{F}, \\ \texttt{o} & : & \mathbf{F} \\ \texttt{dist} & : & \left\{\begin{array}{lcl} \texttt{a} & : & \mathbf{F}, \\ \texttt{o} & : & \mathbf{F} \end{array}\right\} \to \mathbf{F} \end{array}\right\}$ :

$$(\texttt{Obj})\ \frac{(\texttt{Fl})\ \dfrac{}{\heartsuit : \mathcal{P} \vdash 0 : \mathbf{F}} \quad (\texttt{Fl})\ \dfrac{}{\heartsuit : \mathcal{P} \vdash 0 : \mathbf{F}} \quad (\texttt{Abs})\ \dfrac{\dfrac{(\texttt{Call})\ \dfrac{(\texttt{Var})\ \dfrac{}{p : \{\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}\}, \heartsuit : \mathcal{P} \vdash \heartsuit : \mathcal{P}}}{p : \{\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}\}, \heartsuit : \mathcal{P} \vdash \heartsuit.\texttt{a} : \mathbf{F}}}{\ldots}}{\heartsuit : \mathcal{P} \vdash \lambda p.\texttt{rc}\ ((\heartsuit.\texttt{a} - p.\texttt{a})^2 + (\heartsuit.\texttt{o} - p.\texttt{o})^2) : \{\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}\} \to \mathbf{F}}}{\vdash O' : \mathcal{P}}$$

- $O'.\texttt{dist}\ O'$ pas typable

# Types eux-mêmes

- On ajoute un *self* aux types :

$$T ::= \dots \mid \clubsuit$$

- On modifie la règle :

$$(\mathtt{Obj})\ \frac{\begin{array}{c}\Gamma, \heartsuit : \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_n : T_n\} \vdash M_1 : T_1\{\{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_n : T_n\}/\clubsuit\} \\ \dots \qquad \Gamma, \heartsuit : \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_n : T_n\} \vdash M_n : T_n\{\{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_n : T_n\}/\clubsuit\}\end{array}}{\Gamma \vdash [\mathtt{m}_1 : M_1, \dots, \mathtt{m}_n : M_n] : \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_n : T_n\}}$$

quand on ouvre un type objet, $\clubsuit$ est remplacé par le type en question

- exemple on pose $\mathcal{P} = \left\{\begin{array}{lcl} \mathtt{a} & : & \mathbf{F}, \\ \mathtt{o} & : & \mathbf{F} \\ \mathtt{dist} & : & \clubsuit \rightarrow \mathbf{F} \end{array}\right\}$ :

$$(\mathtt{Obj})\ \frac{(\mathtt{Fl})\ \dfrac{}{\heartsuit : \mathcal{P} \vdash 0 : \mathbf{F}} \quad (\mathtt{Fl})\ \dfrac{}{\heartsuit : \mathcal{P} \vdash 0 : \mathbf{F}} \quad (\mathtt{Abs})\ \dfrac{\begin{array}{c} (\mathtt{Call})\ \dfrac{(\mathbf{Var})\ \dfrac{}{p : \mathcal{P}, \heartsuit : \mathcal{P} \vdash \heartsuit : \mathcal{P}}}{p : \mathcal{P}, \heartsuit : \mathcal{P} \vdash \heartsuit.\mathtt{a} : \mathbf{F}} \quad \dots \quad \dfrac{(\mathtt{Call})\ \dfrac{(\mathbf{Var})\ \dfrac{}{p : \mathcal{P}, \heartsuit : \mathcal{P} \vdash p : \mathcal{P}}}{p : \mathcal{P}, \heartsuit : \mathcal{P} \vdash p.\mathtt{o} : \mathbf{F}}}{\dots} \\ \dots \end{array}}{\heartsuit : \mathcal{P} \vdash \lambda p.\mathtt{rc}\ ((\heartsuit.\mathtt{a} - p.\mathtt{a})^2 + (\heartsuit.\mathtt{o} - p.\mathtt{o})^2) : \mathcal{P} \rightarrow \mathbf{F}}}{\vdash O' : \mathcal{P}}$$

# Typage Objet : Inférence

- le système est "moyennement" dirigé par la syntaxe :
  - les étiquettes des attributs apparaissent dans le type pour (**Obj**).
  - quand on type un appel $M.\mathtt{m}$ avec (**Call**) on doit "inventer un type objet pour $M$"
  - solution : on type $M$ et vérifie qu'il contient la bonne méthode. (bof)
  - solution : on peut générer une équation $t_1 = \{\mathtt{m} : T \mid \rho\}$
    - $\rho$ est une variable de rangée
    - $\{\mathtt{m} : T \mid \rho\}$ est le type d'un objet qui contient un attribut $\mathtt{m}$ de type $T$ et potentiellement d'autres attributs.
- Expressivité :
  - on type $O'.\mathtt{dist}\ O'$
  - on ne type pas $O'.\mathtt{dist} \begin{bmatrix} \mathtt{a} & : & 2 \\ \mathtt{o} & : & 2 \end{bmatrix}$
  - on ne type pas

$$O'.\mathtt{dist} \begin{bmatrix} \mathtt{a} & : & 2 \\ \mathtt{o} & : & 2 \\ \mathtt{p} & : & 2 \\ \mathtt{dist} & : & \lambda p.\mathtt{rc}\ ((\heartsuit.\mathtt{a} - p.\mathtt{a})^2 + (\heartsuit.\mathtt{o} - p.\mathtt{o})^2) \end{bmatrix}$$

# Objet : Mutabilité

- les objets définis sont immutables en l'absence de traits impératifs
- par exemple :

$$\mathit{Feu} = \left[ \begin{array}{lcl} \texttt{cou} & : & \text{"vert"} \\ \texttt{nxt} & : & \lambda x.(\texttt{eq\_str}\ \heartsuit.\texttt{cou}\ \text{"vert"}) \left[ \begin{array}{lcl} \texttt{cou} & : & \text{"rouge"} \\ \texttt{nxt} & : & \heartsuit.\texttt{nxt} \end{array} \right] \left[ \begin{array}{lcl} \texttt{cou} & : & \text{"vert"} \\ \texttt{nxt} & : & \heartsuit.\texttt{nxt} \end{array} \right] \end{array} \right]$$

avec eq_str un terme qui compare deux chaînes de caractère et se réduit en $K$ ou $F$

- typé avec :

$$\vdash \mathit{Feu} : \left\{ \begin{array}{lcl} \texttt{cou} & : & \textbf{Str} \\ \texttt{nxt} & : & \bigstar \to \clubsuit \end{array} \right\}$$

- ainsi $\mathit{Feu}.\texttt{nxt}\ ()$ s'évalue en un nouveau feu, avec l'attribut cou à "rouge".

# Objet : Mutabilité (II)

- On peut ajouter une primitive de mise-à-jour :

$$M ::= \ldots \mid M :=_{\mathtt{m}} M \qquad (\mathrm{updg})\,\frac{M \longrightarrow M'}{M :=_{\mathtt{m}} N \longrightarrow M' :=_{\mathtt{m}} N}$$

$$(\mathrm{updd})\,\frac{N \longrightarrow N'}{M :=_{\mathtt{m}} N \longrightarrow M :=_{\mathtt{m}} N'}$$

$$(\mathrm{upd})\,\frac{}{[\mathtt{m}_1 : N_1 \ldots, \mathtt{m}_i : N_i, \ldots, \mathtt{m}_n : N_n] :=_{\mathtt{m}_i} N' \longrightarrow [\mathtt{m}_1 : N_1 \ldots, \mathtt{m}_i : N', \ldots, \mathtt{m}_n : N_n]}$$

$$(\mathtt{Upd})\,\frac{\Gamma \vdash M : \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_i : T_i, \ldots, \mathtt{m}_n : T_n\} \qquad \Gamma \vdash N : T_i}{\Gamma \vdash M :=_{\mathtt{m}_i} N : \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_i : T_i, \ldots, \mathtt{m}_n : T_n\}}$$

- On peut avoir la fonction :

$$\mathtt{suivant} = \lambda f.(\mathsf{eq\_str}\ f.\mathsf{cou}\ \text{"vert"})\ (f :=_{\mathsf{cou}} \text{"rouge"})\ (f :=_{\mathsf{cou}} \text{"vert"})$$

$$\frac{\ldots}{\vdash \mathtt{suivant} : \{\mathtt{cou} : \mathbf{Str}, \mathtt{nxt} : \bigstar \to \clubsuit\} \to \{\mathtt{cou} : \mathbf{Str}, \mathtt{nxt} : \bigstar \to \clubsuit\}}$$

- et la méthode :

$$Feu = \begin{bmatrix} \mathtt{cou} & : & \text{"vert"} \\ \mathtt{nxt} & : & \lambda x.(\mathsf{eq\_str}\ \heartsuit.\mathsf{cou}\ \text{"vert"})\ (\heartsuit :=_{\mathsf{cou}} \text{"rouge"})\ (\heartsuit :=_{\mathsf{cou}} \text{"vert"}) \end{bmatrix}$$

# Objet : Mutabilité (III)

- Pour avoir des objets mutables il faut manipuler des références vers des objets stockés dynamiquement.
- On considère un calcul traits impératifs et une sémantique avec mémoire (cf. Cours 04):

$$M ::= \ldots \mid M :=_{\mathtt{m}} M \qquad (\text{updg})\,\frac{M, \sigma \longrightarrow M', \sigma'}{M :=_{\mathtt{m}} N, \sigma \longrightarrow M' :=_{\mathtt{m}} N, \sigma'}$$

$$(\text{updd})\,\frac{N, \sigma \longrightarrow N', \sigma'}{M :=_{\mathtt{m}} N, \sigma \longrightarrow M :=_{\mathtt{m}} N', \sigma'}$$

$$(\text{upd})\,\frac{\sigma(\xi) = [\mathtt{m}_1 : N_1 \ldots, \mathtt{m}_i : N_i, \ldots, \mathtt{m}_n : N_n]}{\xi :=_{\mathtt{m}_i} N', \sigma \longrightarrow (), \sigma[\xi \leftarrow [\mathtt{m}_1 : N_1 \ldots, \mathtt{m}_i : N', \ldots, \mathtt{m}_n : N_n]]}$$

$$(\text{Upd})\,\frac{\Gamma \vdash M : \textbf{Ref}\ \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_i : T_i, \ldots, \mathtt{m}_n : T_n\} \qquad \Gamma \vdash N : T_i}{\Gamma \vdash M :=_{\mathtt{m}_i} N : \bigstar}$$

- on peut écrire une fonction :

$$\mathtt{suivant} = \lambda f.(\mathtt{eq\_str}\ !f.\mathtt{cou}\ \text{"vert"})\ (f :=_{\mathtt{cou}} \text{"rouge"})\ (f :=_{\mathtt{cou}} \text{"vert"})$$

$$\frac{\ldots}{\vdash \mathtt{suivant} : (\textbf{Ref}\ \{\mathtt{cou} : \textbf{Str}, \mathtt{nxt} : \bigstar \to \clubsuit\}) \to \bigstar}$$

- mais pas une méthode.

# Objet : Mutabilité (IV)

- On peut lier la sémantique des objets aux références :

$$(\texttt{call})\frac{\sigma(\xi) = [\ldots, \texttt{m} : M, \ldots]}{\xi.\texttt{m}, \sigma \longrightarrow M\{\xi/\sigma\}, \sigma} \qquad (\texttt{Call})\frac{\Gamma \vdash M : \textbf{Ref}\ \{\ldots, \texttt{m} : T, \ldots\}}{\Gamma \vdash M.\texttt{m} : T}$$

$$(\texttt{Obj})\frac{\begin{array}{c}\Gamma, \heartsuit : \textbf{Ref}\ \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\} \vdash M_1 : T_1\{\{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\}/\clubsuit\} \\ \ldots \quad \Gamma, \heartsuit : \textbf{Ref}\ \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\} \vdash M_n : T_n\{\{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\}/\clubsuit\}\end{array}}{\Gamma \vdash [\texttt{m}_1 : M_1, \ldots, \texttt{m}_n : M_n] : \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\}}$$

- et définir :

$$\mathit{Feu} = \left[\begin{array}{lll}\texttt{cou} & : & \text{"vert"} \\ \texttt{nxt} & : & \lambda x.(\text{eq\_str}\ \heartsuit.\texttt{cou}\ \text{"vert"})\ (\heartsuit :=_{\texttt{cou}} \text{"rouge"})\ (\heartsuit :=_{\texttt{cou}} \text{"vert"})\end{array}\right]$$

$$(\texttt{Obj})\frac{\ldots}{\vdash \mathit{Feu} : \{\texttt{cou} : \textbf{Str}, \texttt{nxt} : \bigstar \to \bigstar\}}$$

$$\begin{array}{lll} & \texttt{let}\ f := \texttt{ref}\ \mathit{Feu}\ \texttt{in}\ f.\texttt{nxt}\ (), & \emptyset \\ \longrightarrow & \texttt{let}\ f := \xi\ \texttt{in}\ f.\texttt{nxt}\ (), & \xi \leftarrow \mathit{Feu} \\ \longrightarrow^* & (), & \xi \leftarrow [\texttt{cou} : \text{"rouge"}; \texttt{nxt} : \cdots]\end{array}$$

- (cf. $\sigma$-calcul [Abadi-Cardelli 1995])

# Typage Nominal

- on revient aux objets immutables.
- on considère un ensemble de noms de classe : $\mathcal{C}, \mathcal{D}, \ldots$
  - on les ajoute aux types $T ::= \ldots \mid \mathcal{C}$
- un environnement de classe associe des noms de classes à des types (objets). $\Delta = \mathcal{C}_1 : T_1, \ldots, \mathcal{C}_n : T_n$
- le type d'un objet doit être une classe adéquate

$$(\texttt{Obj})\,\frac{\Gamma, \heartsuit : \mathcal{C}\ ;\ \Delta \vdash M_1 : T_1 \qquad \ldots \qquad \Gamma, \heartsuit : \mathcal{C}\ ;\ \Delta \vdash M_n : T_n \qquad \Delta(\mathcal{C}) = \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_n : T_n\}}{\Gamma\ ;\ \Delta \vdash \mathcal{C}}$$

$$(\texttt{Call})\,\frac{\Gamma\ ;\ \Delta \vdash M : \mathcal{C} \qquad \Delta(\mathcal{C}) = \{\texttt{m}_1 : T_1, \ldots, \texttt{m}_i : T_i, \ldots, \texttt{m}_n : T_n\}}{\Gamma\ ;\ \Delta \vdash M.\texttt{m}_i : T_i}$$

- plus besoin de ♣ (définition de classes récursives):

$$O' = \begin{bmatrix} \texttt{a} & : & 0 \\ \texttt{o} & : & 0 \\ \texttt{dist} & : & \lambda p.\texttt{rc}\ ((\heartsuit.\texttt{a} - p.\texttt{a})^2 + (\heartsuit.\texttt{o} - p.\texttt{o})^2) \end{bmatrix}$$

$$(\texttt{Obj})\,\frac{\ldots \qquad \ldots \qquad \dfrac{(\texttt{Call})\,\dfrac{\ldots}{p : \mathcal{P}, \heartsuit : \mathcal{P}\ ;\ \mathcal{P} : [\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}, \texttt{dist} : \mathcal{P} \to \mathbf{F}] \vdash \heartsuit.\texttt{a} : \mathbf{F}}}{\ldots}}{\emptyset\ ;\ \mathcal{P} : [\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}, \texttt{dist} : \mathcal{P} \to \mathbf{F}] \vdash O' : \mathcal{P}}$$

# Typage Nominal : Définition de classe

- on guide le typage avec de nouvelles primitives sans sémantique :

$$M ::= \ldots \mid \mathtt{class}\ \mathcal{C} = \{\mathtt{m}_1 : T, \ldots, \mathtt{m}_n : T\}\ \mathtt{in}\ M \mid \mathtt{new}\ \mathcal{C}\ [\mathtt{m}_1 : M, \ldots, \mathtt{m}_n : M]$$

$$(\mathtt{class})\ \frac{}{\mathtt{class}\ \mathcal{C} = \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_n : T_n\}\ \mathtt{in}\ M \longrightarrow M}$$

$$(\mathtt{new})\ \frac{}{\mathtt{new}\ \mathcal{C}\ [\mathtt{m}_1 : M_1, \ldots, \mathtt{m}_n : M_n] \longrightarrow [\mathtt{m}_1 : M_1, \ldots, \mathtt{m}_n : M_n]}$$

$$(\mathtt{Class})\ \frac{\Gamma\ ;\ \Delta, \mathcal{C} : \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_n : T_n\} \vdash M : T}{\Gamma\ ;\ \Delta \vdash \mathtt{class}\ \mathcal{C} = \{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_n : T_n\}\ \mathtt{in}\ M : T\{\{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_n : T_n\}/\mathcal{C}\}}$$

$$(\mathtt{New})\ \frac{\Gamma; \Delta \vdash [\mathtt{m}_1 : M_1, \ldots, \mathtt{m}_n : M_n] : \mathcal{C}}{\Gamma; \Delta \vdash \mathtt{new}\ \mathcal{C}\ [\mathtt{m}_1 : M_1, \ldots, \mathtt{m}_n : M_n] : \mathcal{C}}$$

- ces primitives servent uniquement à annoter le code avec des noms de classes.
- dans le calcul à objets mutables, on maintient une bibliothèque de classes pendant les réductions.

# Typage Nominal : Définition de classe (II)

- exemple :

$$
\begin{array}{lll}
Pr & = & \texttt{class}\ \mathcal{P} = \{\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}, \texttt{dist} : \mathcal{P} \to \mathbf{F}\} \\
 & & \texttt{in let}\ o = \texttt{new}\ \mathcal{P} \left[\begin{array}{lll} \texttt{a} & : & 0 \\ \texttt{o} & : & 0 \\ \texttt{dist} & : & \lambda p.\texttt{rc}\ ((\heartsuit.\texttt{a} - p.\texttt{a})^2 + (\heartsuit.\texttt{o} - p.\texttt{o})^2) \end{array}\right] \\
 & & \texttt{in let}\ p_1 = \texttt{new}\ \mathcal{P} \left[\begin{array}{lll} \texttt{a} & : & 2 \\ \texttt{o} & : & 3 \\ \texttt{dist} & : & \lambda p.\texttt{rc}\ ((\heartsuit.\texttt{a} - p.\texttt{a})^2 + (\heartsuit.\texttt{o} - p.\texttt{o})^2) \end{array}\right] \\
 & & \texttt{in}\ o.\texttt{dist}\ p_1
\end{array}
$$

$$
Pr \longrightarrow^* \mathbf{rc}\ ((0-2)^2 + (0-3)^2) \longrightarrow^* \ldots
$$

$$
(\textbf{Class})\ \dfrac{\ldots \qquad \dfrac{\dfrac{\ldots}{o : \mathcal{P}, p_1 : \mathcal{P}, \heartsuit : \mathcal{P}, p : \mathcal{P}\ ;\ \mathcal{P} : \{\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}, \texttt{dist} : \mathcal{P} \to \mathbf{F}\} \vdash \heartsuit.\texttt{a} : \mathbf{F}}}{\ldots} \qquad \ldots}{\vdash Pr : \mathbf{F}}
$$

- on type `dist` deux fois (une fois dans chaque `new`).

# Typage Nominal : Constructeur

- on voudrait remonter les méthodes générales dans la classe
- et ne laisser que les attributs particuliers dans le `new`

$$J ::= T \mid M$$

$$M ::= \dots \mid \texttt{class}\ \mathcal{C} = \{\texttt{m}_1 : J, \dots, \texttt{m}_n : J\}\ \texttt{in}\ M \mid \texttt{new}\ \mathcal{C}\ [\texttt{m}_1 : M, \dots, \texttt{m}_n : M]$$

$$(\texttt{class})\ \frac{}{\begin{array}{c}\texttt{class}\ \mathcal{C} = \{\texttt{m}_1 : T_1, \dots, \texttt{m}_k : T_k, \texttt{m}_{k+1} : M_{k+1} \dots, \texttt{m}_n : M_n\}\ \texttt{in}\ M \\ \longrightarrow M\{\texttt{new}\ \mathcal{C}\ [\texttt{m}_1 : M_1, \dots, \texttt{m}_n : M_n]/\texttt{new}\ \mathcal{C}\ [\texttt{m}_1 : M_1, \dots, \texttt{m}_k : M_k]\}\end{array}}$$

$$(\texttt{new})\ \frac{}{\texttt{new}\ \mathcal{C}\ [\texttt{m}_1 : M_1, \dots, \texttt{m}_n : M_n] \longrightarrow [\texttt{m}_1 : M_1, \dots, \texttt{m}_n : M_n]}$$

$$(\texttt{Class})\ \frac{\begin{array}{c}\Gamma, \heartsuit : \mathcal{C}\ ;\ \Delta, \mathcal{C} : \{\texttt{m}_1 : T_1, \dots, \texttt{m}_n : T_n\} \vdash M_{k+1} : T_{k+1} \\ \dots \qquad \Gamma, \heartsuit : \mathcal{C}\ ;\ \Delta, \mathcal{C} : \{\texttt{m}_1 : T_1, \dots, \texttt{m}_n : T_n\} \vdash M_n : T_n \\ \Gamma\ ;\ \Delta, \mathcal{C} : \{\texttt{m}_1 : T_1, \dots, \texttt{m}_n : T_n\} \vdash M : T\end{array}}{\begin{array}{c}\Gamma\ ;\ \Delta \vdash \texttt{class}\ \mathcal{C} = \{\texttt{m}_1 : T_1, \dots, \texttt{m}_k : T_k, \texttt{m}_{k+1} : M_{k+1}, \dots, \texttt{m}_n : M_n\} \\ \texttt{in}\ M : T\{\{\texttt{m}_1 : T_1, \dots, \texttt{m}_n : T_n\}/\mathcal{C}\}\end{array}}$$

$$(\texttt{New})\ \frac{\Gamma, \heartsuit : \mathcal{C}; \Delta \vdash \texttt{m}_1 : M_1 \qquad \dots \qquad \Gamma, \heartsuit : \mathcal{C}; \Delta \vdash \texttt{m}_k : M_k \qquad \Delta(\mathcal{C}) = \{\texttt{m}_1 : T_1, \dots, \texttt{m}_n : T_n\}}{\Gamma; \Delta \vdash \texttt{new}\ \mathcal{C}\ [\texttt{m}_1 : M_1, \dots, \texttt{m}_n : M_n] : \mathcal{C}}$$

SORBONNE UNIVERSITÉ

# Constructeur : Exemple

- exemple :

$$
\begin{array}{lcl}
Pr & = & \texttt{class}\ \mathcal{P} = \{\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}, \texttt{dist} : \lambda p.\texttt{rc}\ ((\heartsuit.\texttt{a} - p.\texttt{a})^2 + (\heartsuit.\texttt{o} - p.\texttt{o})^2)\} \\
 & & \texttt{in let}\ o = \texttt{new}\ \mathcal{P} \begin{bmatrix} \texttt{a} & : & 0 \\ \texttt{o} & : & 0 \end{bmatrix} \\
 & & \texttt{in let}\ p_1 = \texttt{new}\ \mathcal{P} \begin{bmatrix} \texttt{a} & : & 2 \\ \texttt{o} & : & 3 \end{bmatrix} \\
 & & \texttt{in}\ o.\texttt{dist}\ p_1
\end{array}
$$

$$
Pr \longrightarrow^* \mathbf{rc}\ ((0-2)^2 + (0-3)^2) \longrightarrow^* \ldots
$$

$$
(\mathbf{Class})\ \dfrac{\ldots \qquad \dfrac{\dfrac{\ldots}{o : \mathcal{P}, p_1 : \mathcal{P}, \heartsuit : \mathcal{P}, p : \mathcal{P}\ ;\ \mathcal{P} : \{\texttt{a} : \mathbf{F}, \texttt{o} : \mathbf{F}, \texttt{dist} : \mathcal{P} \to \mathbf{F}\} \vdash \heartsuit.\texttt{a} : \mathbf{F}}}{\ldots} \qquad \ldots}{\vdash Pr : \mathbf{F}}
$$

- on type `dist` une seule fois dans la classe.

# Sous-Typage

- "$S$ sous-type de $T$" ($S \leq T$):
  - vision ensembliste : $S \subseteq T$
  - "tous les $S$ sont des $T$"
  - peut-être certains $T$ ne sont pas des $S$ (sous-typage strict)
- subsomption :
  - "partout où on a besoin d'un $T$, on peut utiliser un $S$"
  - vision sémantique du sous-typage
- en objet :
  - $S \leq T$ : tous les attributs de $T$ sont dans $S$
  - $S$ peut avoir d'autres attributs.
- intérêt :
  - regrouper des objets de types différents qui peuvent être utilisés par le même programme.
  - c'est un polymorphisme.

# Sous-Typage Nominal

- mécanisme d'héritage :

$$M ::= \dots \mid \texttt{subclass}\ \mathcal{C}_1\ \texttt{of}\ \mathcal{C}_2 = \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_n : T_n\}\ \texttt{in}\ M$$

$$(\textbf{class})\ \frac{}{\begin{array}{c}\texttt{class}\ \mathcal{C} = \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_k : T_k, \mathtt{m}_{k+1} : M_{k+1} \dots, \mathtt{m}_n : M_n\}\ \texttt{in}\ M \\ \longrightarrow M\{\texttt{new}\ \mathcal{C}\ [\mathtt{m}_1 : M_1, \dots, \mathtt{m}_n : M_n]/\texttt{new}\ \mathcal{C}\ [\mathtt{m}_1 : M_1, \dots, \mathtt{m}_k : M_k]\} \\ \{\texttt{class}\ \mathcal{C}' = \{\mathtt{m}_1 : J_1, \dots, \mathtt{m}_n : J_n, \mathtt{m}_{n+1} : J_{n+1}, \dots, \mathtt{m}_l : J_l\}\ \texttt{in}\ M' \\ /\texttt{subclass}\ \mathcal{C}'\ \texttt{of}\ \mathcal{C} = \{\mathtt{m}_{n+1} : J_{n+1}, \dots, \mathtt{m}_l : J_l\}\ \texttt{in}\ M'\}\end{array}}$$

$$(\textbf{Subclass})\ \frac{\begin{array}{c}\Gamma, \heartsuit : \mathcal{C}'\ ;\ \Delta, \mathcal{C}' : \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_l : T_l\}\ ;\ \Theta, \mathcal{C}' < \mathcal{C} \vdash M_{n+k+1} : T_{n+k+1} \\ \dots \qquad \Gamma, \heartsuit : \mathcal{C}'\ ;\ \Delta, \mathcal{C}' : \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_l : T_l\}\ ;\ \Theta, \mathcal{C}' < \mathcal{C} \vdash M_l : T_l \\ \Gamma\ ;\ \Delta, \mathcal{C}' : \{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_l : T_l\}; \Theta, \mathcal{C}' < \mathcal{C} \vdash M : T \qquad \Delta(\mathcal{C}) = \{\mathtt{m}_1 : J_1, \dots, \mathtt{m}_n : J_n\}\end{array}}{\begin{array}{c}\Gamma\ ;\ \Delta\ ;\ \Theta \vdash \texttt{subclass}\ \mathcal{C}'\ \texttt{of}\ \mathcal{C} = \{\mathtt{m}_{n+1} : T_{n+1}, \dots, \mathtt{m}_{n+k} : T_{n+k}, \mathtt{m}_{n+k+1} : M_{n+k+1}, \dots, \mathtt{m}_l : M_l\} \\ \texttt{in}\ M : T\{\{\mathtt{m}_1 : T_1, \dots, \mathtt{m}_l : T_l\}/\mathcal{C}'\}\end{array}}$$

$$(\textbf{Subsum})\ \frac{\Gamma\ ;\ \Delta\ ;\ \Theta \vdash M : \mathcal{C}' \qquad \mathcal{C}' < \mathcal{C} \in \Theta}{\Gamma\ ;\ \Delta\ ;\ \Theta \vdash M : \mathcal{C}}$$

- contexte d'héritage $\Theta = \mathcal{C} < \mathcal{C}', \dots$
- on remplace dynamiquement les `subclass` en *inlinant* la classe complète.
- le typage permet de surtyper un objet.

# Sous-Typage Nominal : Exemple

- exemple :

$$
\begin{array}{lcl}
Pr & = & \texttt{class}\ \mathcal{P} = \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}, \mathtt{dist} : \lambda p.\mathtt{rc}\ ((\heartsuit.\mathtt{a} - p.\mathtt{a})^2 + (\heartsuit.\mathtt{o} - p.\mathtt{o})^2)\} \\
 & & \texttt{subclass}\ \mathcal{Q}\ \texttt{of}\ \mathcal{P} = \{\mathtt{c} : \mathbf{Str}\} \\
 & & \texttt{in let}\ o = \texttt{new}\ \mathcal{P} \begin{bmatrix} \mathtt{a} & : & 0 \\ \mathtt{o} & : & 0 \end{bmatrix} \\
 & & \texttt{in let}\ p_1 = \texttt{new}\ \mathcal{Q} \begin{bmatrix} \mathtt{a} & : & 2 \\ \mathtt{o} & : & 3 \\ \mathtt{c} & : & \text{"bleu"} \end{bmatrix} \\
 & & \texttt{in}\ o.\texttt{dist}\ p_1
\end{array}
$$

$$
Pr \longrightarrow^* \mathtt{rc}\ ((0-2)^2 + (0-3)^2) \longrightarrow^* \ldots
$$

$$
(\textbf{Class})\ \dfrac{\ldots \quad (\textbf{Subsum})\ \dfrac{(\textbf{Var})\ \dfrac{}{o : \mathcal{P}, p_1 : \mathcal{Q}\ ;\ \mathcal{P} : \{\mathtt{a} : \mathtt{F}, \mathtt{o} : \mathtt{F}, \mathtt{dist} : \mathcal{C} \longrightarrow \mathbf{F}\}, \mathcal{Q} : \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}, \mathtt{dist} : \mathcal{C} \longrightarrow \mathbf{F}, \mathtt{c} : \mathbf{Str}\}\ ;\ \mathcal{Q} < \mathcal{P} \vdash p_1 : \mathcal{Q}}}{o : \mathcal{P}, p_1 : \mathcal{Q}\ ;\ \mathcal{P} : \{\mathtt{a} : \mathtt{F}, \mathtt{o} : \mathtt{F}, \mathtt{dist} : \mathcal{C} \longrightarrow \mathbf{F}\}, \mathcal{Q} : \{\mathtt{a} : \mathbf{F}, \mathtt{o} : \mathbf{F}, \mathtt{dist} : \mathcal{C} \longrightarrow \mathbf{F}, \mathtt{c} : \mathbf{Str}\}\ ;\ \mathcal{Q} < \mathcal{P} \vdash p_1 : \mathcal{P}} \quad \ldots}{\vdash Pr : \mathbf{F}}
$$

- 
  - `subclass` ajoute dans $\Theta$ que les $\mathcal{Q}$ sont des $\mathcal{P}$.
  - l'argument de `dist` doit être un $\mathcal{P}$
  - la règle (**Subsum**) nous autorise donc à montrer que cet argument est un $\mathcal{Q}$.
- si on veut se passer des classes (et appeler distance depuis un point avec un point coloré comme argument)

SORBONNE UNIVERSITÉ

# Typage Structurel

- on revient aux objets basiques (sans les classes).
- $\rho$ variable de rangée
  - $[\mathtt{m}_1 : M_1, \ldots, \mathtt{m}_n : M_n \mid \rho]$ : un objet qui contient, entre autres, les méthodes $\mathtt{m}_1, \ldots, \mathtt{m}_n$.
- $$T ::= \ldots \mid \left\{ \begin{array}{ccc} \mathtt{m}_1 & : & T, \\ \mathtt{m}_2 & : & T, \\ \ldots & & \\ \mathtt{m}_n & : & T \\ & \mid & \rho \end{array} \right\} \mid \forall \alpha . T \mid \forall \rho . T \qquad (\mathtt{Call}) \frac{\Gamma \vdash M : \{\mathtt{m} : T | \rho\}}{\Gamma \vdash M.\mathtt{m} : T}$$

  $$(\mathtt{InstO}) \frac{\Gamma \vdash M : \forall \rho . T}{\Gamma \vdash M : T\{\mathtt{m}_1 : T_1, \ldots, \mathtt{m}_n : T_n / \rho\}}$$

- on généralise aussi les variables de rangée dans la règle (**Let**)
- quand on appelle une méthode, on requiert uniquement que l'objet ait un type qui contienne la méthode (avec le bon type).
- l'instantiation permet de compléter une rangée.

# Typage Structurel : Exemple

- exemple :

$$
\begin{array}{rcl}
Pr & = & \texttt{let}\ o = \begin{bmatrix} \mathrm{a} & : & 0 \\ \mathrm{o} & : & 0 \end{bmatrix} \\
 & & \texttt{in let}\ p_1 = \begin{bmatrix} \mathrm{a} & : & 2 \\ \mathrm{o} & : & 3 \\ \mathrm{c} & : & \text{"bleu"} \end{bmatrix} \\
 & & \texttt{in let dist} = \lambda p, p'.\texttt{rc}\ ((p'.\mathrm{a} - p.\mathrm{a})^2 + (p'.\mathrm{o} - p.\mathrm{o})^2) \\
 & & \texttt{in dist}\ o\ p_1
\end{array}
$$

$$
Pr \longrightarrow^* \texttt{rc}\ ((0-2)^2 + (0-3)^2) \longrightarrow^* \ldots
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{???}{\Gamma, p : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}, \rho_1\}, p' : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}, \rho_2\} \vdash p' : \{\mathrm{a} : \mathbf{F} \mid \rho_0\}}}{\Gamma, p : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}, \rho_1\}, p' : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}, \rho_2\} \vdash p'.\mathrm{a} : \mathbf{F}}}{\ldots} \quad \dfrac{\dfrac{\ldots}{\Gamma \vdash \texttt{dist} : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}\} \to \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}, \mathrm{c} : \mathbf{Str}\} \to \mathbf{F}}}{\ldots}}{o : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}\}, p_1 : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F}, \mathrm{c} : \mathbf{Str}\}, \texttt{dist} : \forall \rho_1, \rho_2.(\{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F} \mid \rho_1\} \to \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F} \mid \rho_2\} \to \mathbf{F}) \vdash \texttt{dist}\ o\ p_1 : \mathbf{F}}}{\dfrac{\ldots}{\vdash Pr : \mathbf{F}}}
$$

- problème :
  - on veut montrer que $p' : \{\mathrm{a} : \mathbf{F} \mid \rho_0\}$
  - on sait que que $p' : \{\mathrm{a} : \mathbf{F}, \mathrm{o} : \mathbf{F} \mid \rho_2\}$
  - l'instantiation n'est pas utile.

# Sous-Typage Structurel

- on définit la relation $\leq$ sur les types :

$$\frac{}{T \leq T}$$

$$\frac{T_1 \leq T'_1 \qquad \dots \qquad T_k \leq T'_k}{\{m_1 : T_1, \dots, m_k : T_k, m_{k+1} : T_{k+1}, \dots, M_n : T_n\} \leq \{m_1 : T'_1, \dots, m_k : T'_k\}}$$

$$\frac{T_1 \leq T'_1 \qquad \dots \qquad T_k \leq T'_k}{\{m_1 : T_1, \dots, m_k : T_k, m_{k+1} : T_{k+1}, \dots, M_n : T_n \mid \rho\} \leq \{m_1 : T'_1, \dots, m_k : T'_k \mid \rho\}}$$

$$\frac{T'_a \leq T_a \qquad T_r \leq T'_r}{T_a \rightarrow T_r \leq T'_a \rightarrow T'_r}$$

- et la subsomption :

$$(\textbf{Subsum})\frac{\Gamma \vdash M : T \qquad T \leq T'}{\Gamma \vdash M : T'}$$

- Pour les fonctions, on est covariant à droite de la flèche et contravariant à gauche.

SORBONNE UNIVERSITÉ

# Sous-Typage Structurel : Exemple

- exemple :

$$
\begin{array}{lll}
Pr & = & \texttt{let } o = \begin{bmatrix} \mathsf{a} & : & 0 \\ \mathsf{o} & : & 0 \end{bmatrix} \\
 & & \texttt{in let } p_1 = \begin{bmatrix} \mathsf{a} & : & 2 \\ \mathsf{o} & : & 3 \\ \mathsf{c} & : & \text{"bleu"} \end{bmatrix} \\
 & & \texttt{in let dist} = \lambda p, p'.\texttt{rc}\ ((p'.\mathsf{a} - p.\mathsf{a})^2 + (p'.\mathsf{o} - p.\mathsf{o})^2) \\
 & & \texttt{in dist } o\ p_1
\end{array}
$$

$$
Pr \longrightarrow^* \texttt{rc}\ ((0-2)^2 + (0-3)^2) \longrightarrow^* \ldots \qquad \frac{\ldots}{\{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \rho_1\} \leq \{\mathsf{a} : \mathbf{F}, \rho_1\}}
$$

$$
\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{\Gamma, p : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \rho_1\}, p' : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \rho_2\} \vdash p' : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F} \mid \rho_2\}}\,(\texttt{Var})}{\Gamma, p : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \rho_1\}, p' : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \rho_2\} \vdash p' : \{\mathsf{a} : \mathbf{F} \mid \rho_2\}}\,(\texttt{Subsum})}{\Gamma, p : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \rho_1\}, p' : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \rho_2\} \vdash p'.\mathsf{a} : \mathbf{F}}\,(\texttt{Call}) \quad \cfrac{\ldots}{\Gamma \vdash \texttt{dist} : \ldots}}{\ldots \qquad\qquad \ldots}}{o : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}\}, p_1 : \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F}, \mathsf{c} : \mathbf{Str}\}, \texttt{dist} : \forall \rho_1, \rho_2.(\{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F} \mid \rho_1\} \to \{\mathsf{a} : \mathbf{F}, \mathsf{o} : \mathbf{F} \mid \rho_2\} \to \mathbf{F}) \vdash \texttt{dist } o\ p_1 : \mathbf{F}}}{\cfrac{\ldots}{\vdash Pr : \mathbf{F}}}
$$

# Conclusion

- TDs 03-06 - Travail en autonomie
  - réalisation d'un évaluateur-typeur,
  - synthèse d'article.
- Cours 06 : Surcharge, Typage moderne.