

Jean Goubault-Larrecq

Logique classique propositionnelle

31 mars 2020

Aujourd'hui

- ❖ Suite de notre étude du λ -calcul simplement typé:
logique **classique**
- ❖ ... c'est tout! Ca va nous occuper bien assez, allez...

Logique classique

La logique propositionnelle classique

- ❖ Juste deux opérateurs ici: \Rightarrow et \perp
- ❖ En classique (pas en intuitionniste), on peut définir les autres (\wedge , \vee , \neg) à partir de ces deux-là:
 - $\neg F = F \Rightarrow \perp$
 - $F \vee G = (\neg F) \Rightarrow G$
 - $F \wedge G = \neg(F \Rightarrow \neg G)$
- ❖ mais on pourrait aussi utiliser les règles de la dernière fois

Rappelez-vous: logique minimale+faux

❖ Types (formules):

$F, G, \dots ::= A$

| $F \Rightarrow G$

| \perp (« faux »)

❖ Termes:

$u, v, \dots ::= x$

| uv

| $\lambda x.u$

| ∇u

❖ Réduction:

(β) $(\lambda x.u)v \rightarrow u[x:=v]$

(∇) $(\nabla u)v \rightarrow \nabla u$

du faux on
peut déduire
n'importe quoi

$$\frac{\Gamma \vdash u : \perp}{\Gamma \vdash \nabla u : G} \quad (\perp E)$$

élimination du faux
(... et pas d'intro du faux!)

$$\frac{}{\Gamma, x:F \vdash x : F} \quad (\lambda x)$$

$$\frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} \quad (\Rightarrow E)$$

$$\frac{\Gamma, x:F \vdash u : G}{\Gamma \vdash \lambda x.u : F \Rightarrow G} \quad (\Rightarrow I)$$

... logique minimale + faux \rightarrow ...

- ❖ Types (formules):
 - $F, G, \dots ::= A$
 - $| F \Rightarrow G$
 - $| \perp$ (« faux »)
- ❖ Termes:
 - $u, v, \dots ::= x$
 - $| uv$
 - $| \lambda x.u$
 - $| Cu$
- ❖ Réduction:
 - $(\beta) (\lambda x.u)v \rightarrow u[x:=v]$
 - $(\nabla) (\nabla u)v \rightarrow \nabla u$

$$\frac{\Gamma \vdash u : \perp}{\Gamma \vdash \nabla u : G} \text{ (}\perp\text{E)}$$

$$\frac{}{\Gamma, x:F \vdash x : F} \text{ (}\Lambda x\text{)}$$

$$\frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} \text{ (}\Rightarrow\text{E)}$$

$$\frac{\Gamma, x:F \vdash u : G}{\Gamma \vdash \lambda x.u : F \Rightarrow G} \text{ (}\Rightarrow\text{I)}$$

... \rightarrow logique classique

❖ Types (formules):

$F, G, \dots ::= A$

| $F \Rightarrow G$

| \perp (« faux »)

❖ Termes:

$u, v, \dots ::= x$

| uv

| $\lambda x.u$

| Cu

❖ Réduction:

(β) $(\lambda x.u)v \rightarrow u[x:=v]$

(∇) $(\nabla u)v \rightarrow \nabla u$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

$$\frac{}{\Gamma, x:F \vdash x : F} (\Lambda x)$$

$$\frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} (\Rightarrow E)$$

$$\frac{\Gamma, x:F \vdash u : G}{\Gamma \vdash \lambda x.u : F \Rightarrow G} (\Rightarrow I)$$

... \rightarrow logique classique

- ❖ Types (formules):
 - $F, G, \dots ::= A$
 - | $F \Rightarrow G$
 - | \perp (« faux »)
- ❖ Termes:
 - $u, v, \dots ::= x$
 - | uv
 - | $\lambda x.u$
 - | **Cu**
- ❖ Réduction:
 - (β) $(\lambda x.u)v \rightarrow u[x:=v]$
 - (∇) $(\nabla u)v \rightarrow \nabla u$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \quad (\neg\neg E)$$

élimination de
la double négation (!)

$$(\neg\neg G = (G \Rightarrow \perp) \Rightarrow \perp)$$

(Ax)

$$\frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} \quad (\Rightarrow E)$$

$$\frac{\Gamma, x:F \vdash u : G}{\Gamma \vdash \lambda x.u : F \Rightarrow G} \quad (\Rightarrow I)$$

... \rightarrow logique classique

❖ Types (formules):

$F, G, \dots ::= A$

| $F \Rightarrow G$

| \perp (« faux »)

❖ Termes:

$u, v, \dots ::= x$

| uv

| $\lambda x.u$

| Cu

❖ Réduction:

(β) $(\lambda x.u)v \rightarrow u[x:=v]$

(∇) $(\nabla u)v \rightarrow \nabla u$

raisonnement
par l'absurde

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \quad (\neg\neg E)$$

élimination de
la double négation (!)

$(\neg\neg G = (G \Rightarrow \perp) \Rightarrow \perp)$

(Ax)

$$\frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} \quad (\Rightarrow E)$$

$$\frac{\Gamma, x:F \vdash u : G}{\Gamma \vdash \lambda x.u : F \Rightarrow G} \quad (\Rightarrow I)$$

... \rightarrow logique classique

❖ Types (formules):

$F, G, \dots ::= A$

| $F \Rightarrow G$

| \perp (« faux »)

❖ Termes:

$u, v, \dots ::= x$

| uv

| $\lambda x.u$

| Cu

❖ Réduction:

(β) $(\lambda x.u)v \rightarrow u[x:=v]$

(**C**) $(Cu)v \rightarrow ?$

raisonnement
par l'absurde

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \quad (\neg\neg E)$$

élimination de
la double négation (!)

$(\neg\neg G = (G \Rightarrow \perp) \Rightarrow \perp)$

(Ax)

$$\frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} \quad (\Rightarrow E)$$

$$\frac{\Gamma, x:F \vdash u : G}{\Gamma \vdash \lambda x.u : F \Rightarrow G} \quad (\Rightarrow I)$$

Raisonnement par l'absurde

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash \mathbf{C}u : G} \text{ } (\neg\neg\text{E})$$

- ❖ Avant de découvrir la règle $(\mathbf{C}u)v \rightarrow ?$, parlons de $(\neg\neg\text{E})$

Raisonnement par l'absurde

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

- ❖ Avant de découvrir la règle $(Cu)v \rightarrow ?$, parlons de $(\neg\neg E)$
- ❖ « pour prouver G , supposons $\neg G$, et déduisons \perp »
→ raisonnement **par l'absurde**

Raisonnement par l'absurde

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

- ❖ Avant de découvrir la règle $(Cu)v \rightarrow ?$, parlons de $(\neg\neg E)$
- ❖ « pour prouver G , supposons $\neg G$, et déduisons \perp »
→ raisonnement **par l'absurde**
- ❖ différent de:
« pour prouver $\neg G$, supposons G , et déduisons \perp »
→ ... qui est prouvable en intuitionniste
(rappel: $\neg G = G \Rightarrow \perp$, et utilisez $(\Rightarrow I)$)

La règle $(Cu)v \rightarrow \dots$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$

- ❖ Je vous propose de tenter de trouver ‘...’ en cherchant un terme du même type que le côté gauche
- ❖ On verra bien ce que ça donne!

La règle $(Cu)v \rightarrow \dots$

❖ Si on veut typer le côté gauche...

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

(C) $(Cu)v \rightarrow$?

La règle $(Cu)v \rightarrow \dots$

❖ Si on veut typer le côté gauche...

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$

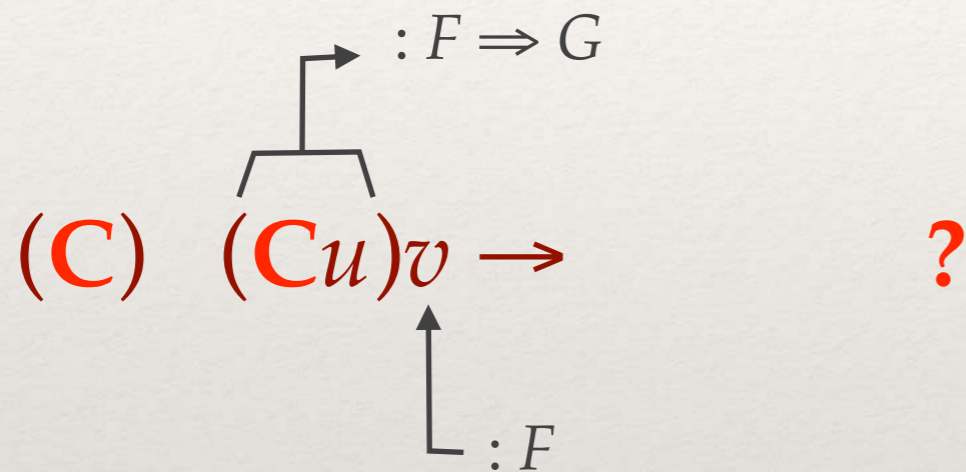
$$(C) \quad (Cu)v \rightarrow \quad ?$$

\uparrow
 $\lfloor : F$

La règle $(Cu)v \rightarrow \dots$

❖ Si on veut typer le côté gauche...

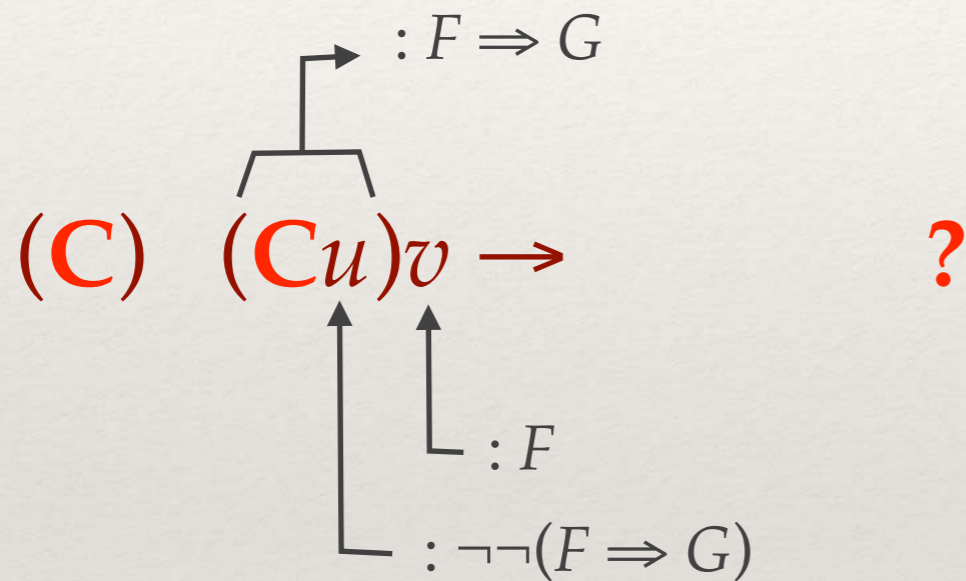
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

❖ Si on veut typer le côté gauche...

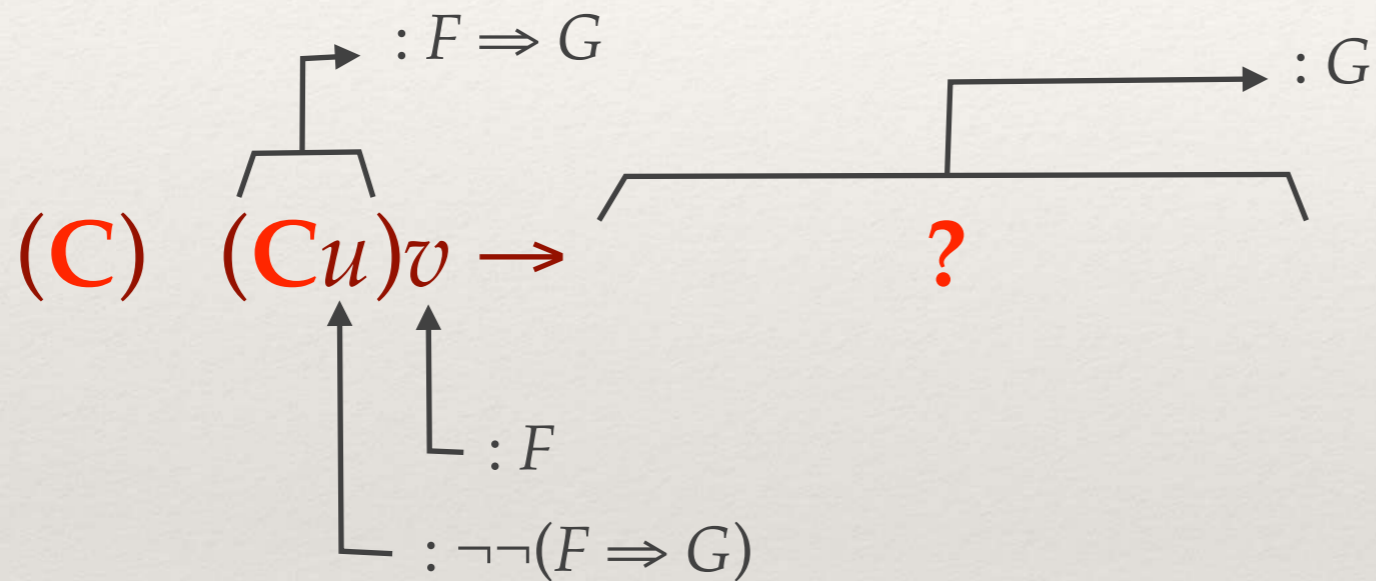
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

- ❖ Si on veut typer le côté gauche...

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

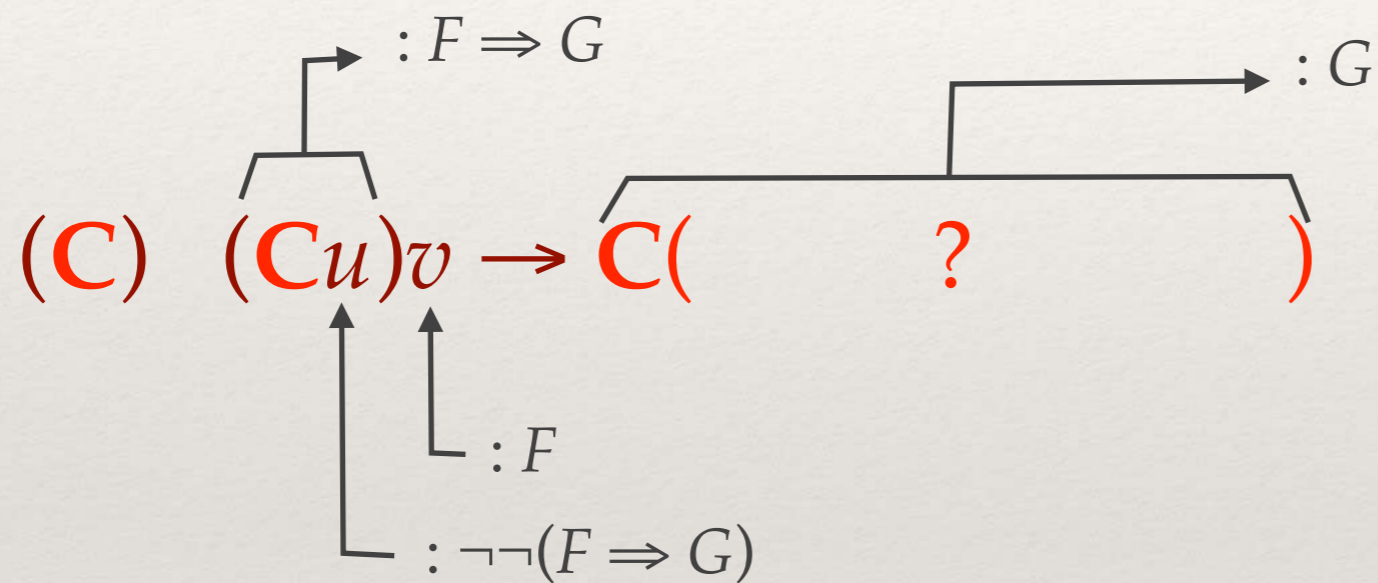


- ❖ alors le côté droit doit être de type G

La règle $(Cu)v \rightarrow \dots$

❖ Cherchons à faire remonter **C**:

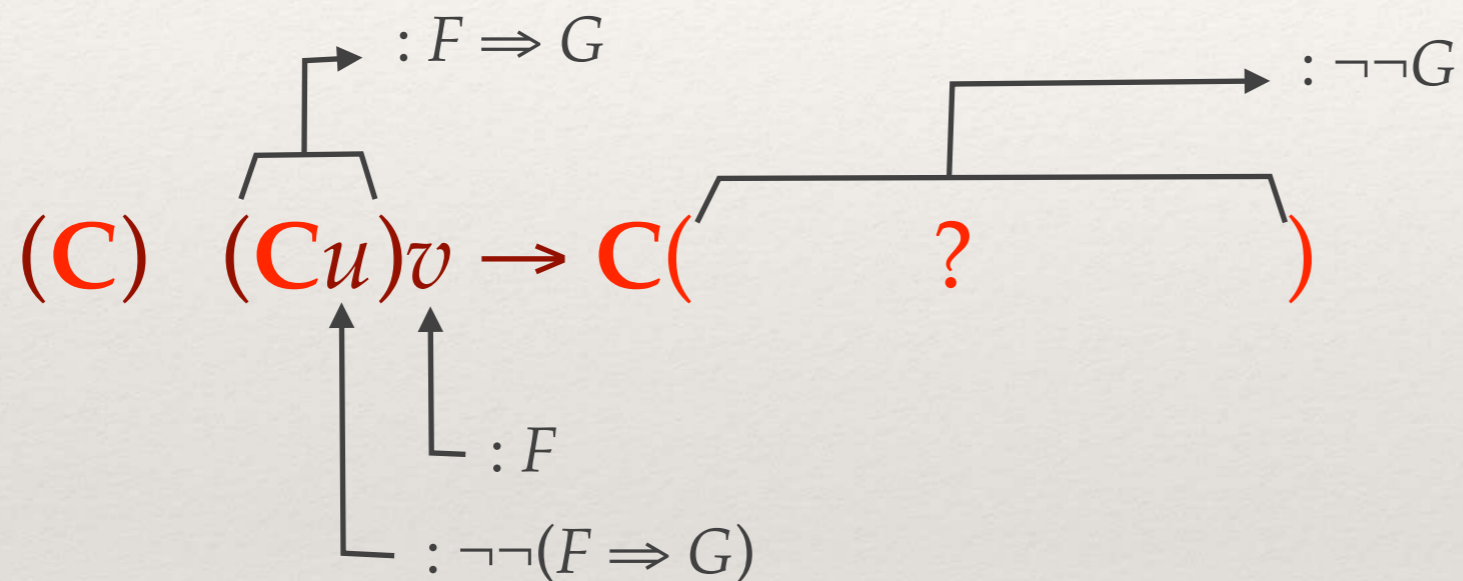
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

❖ Cherchons à faire remonter \mathbf{C} :

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

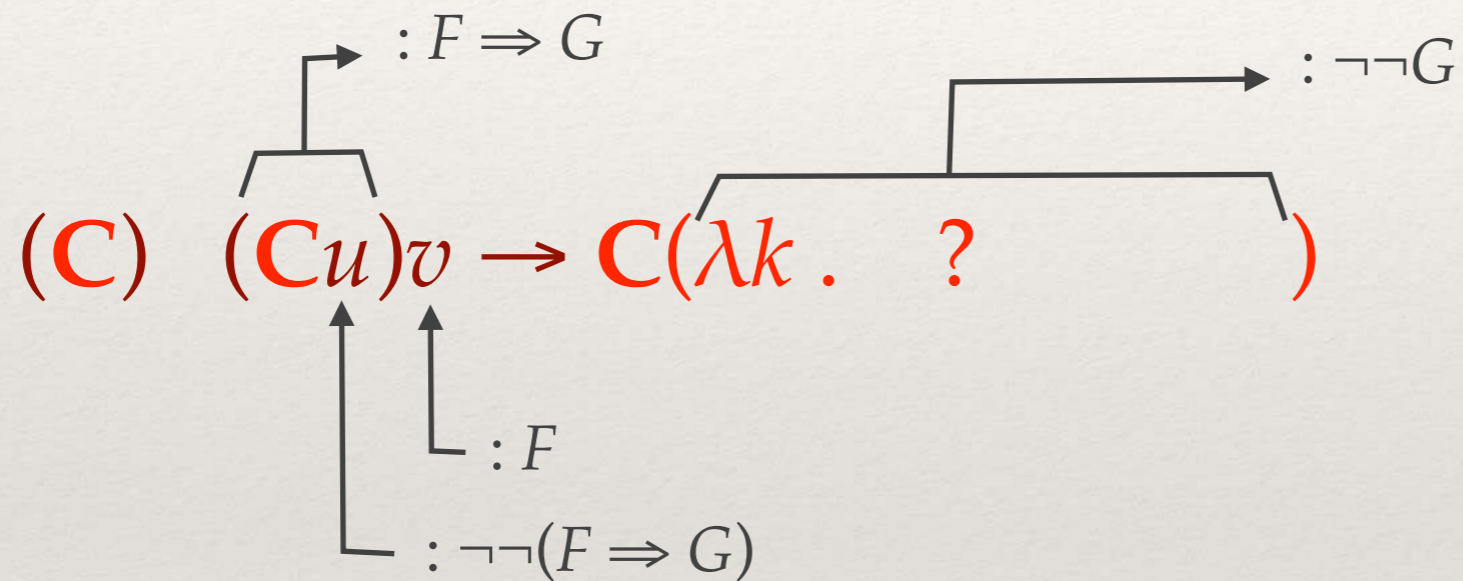


❖ donc le terme sous \mathbf{C} doit être de type $\neg\neg G$

La règle $(Cu)v \rightarrow \dots$

❖ Introduisons une abstraction

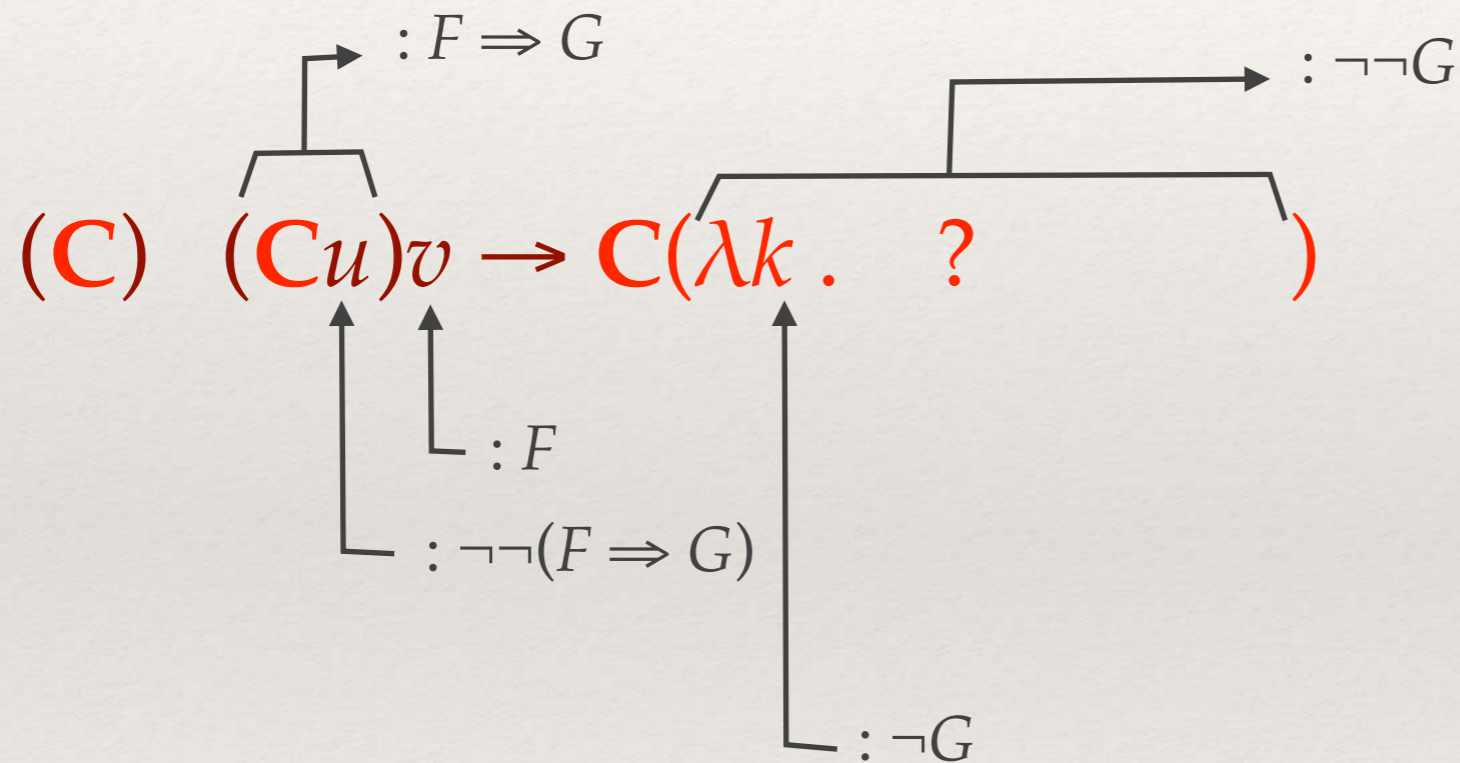
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

❖ Introduisons une abstraction

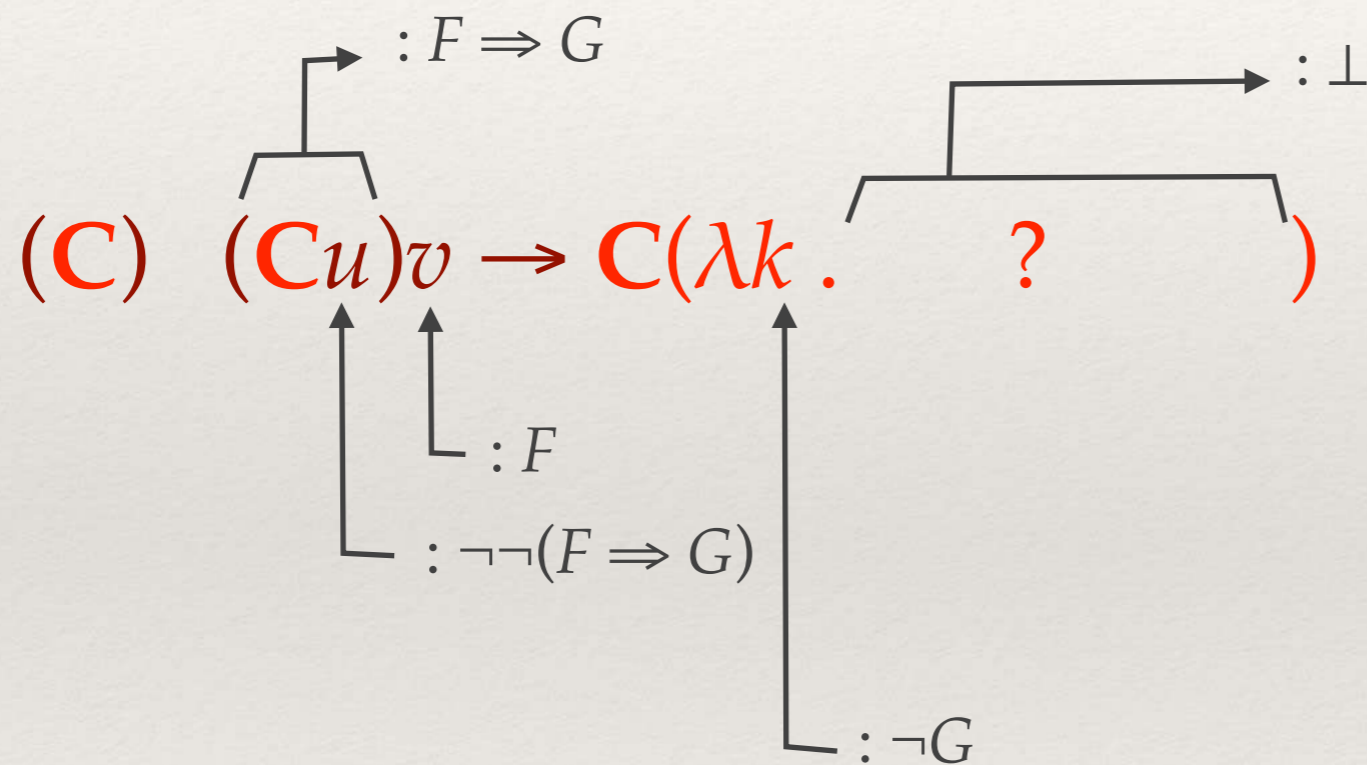
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

❖ Introduisons une abstraction

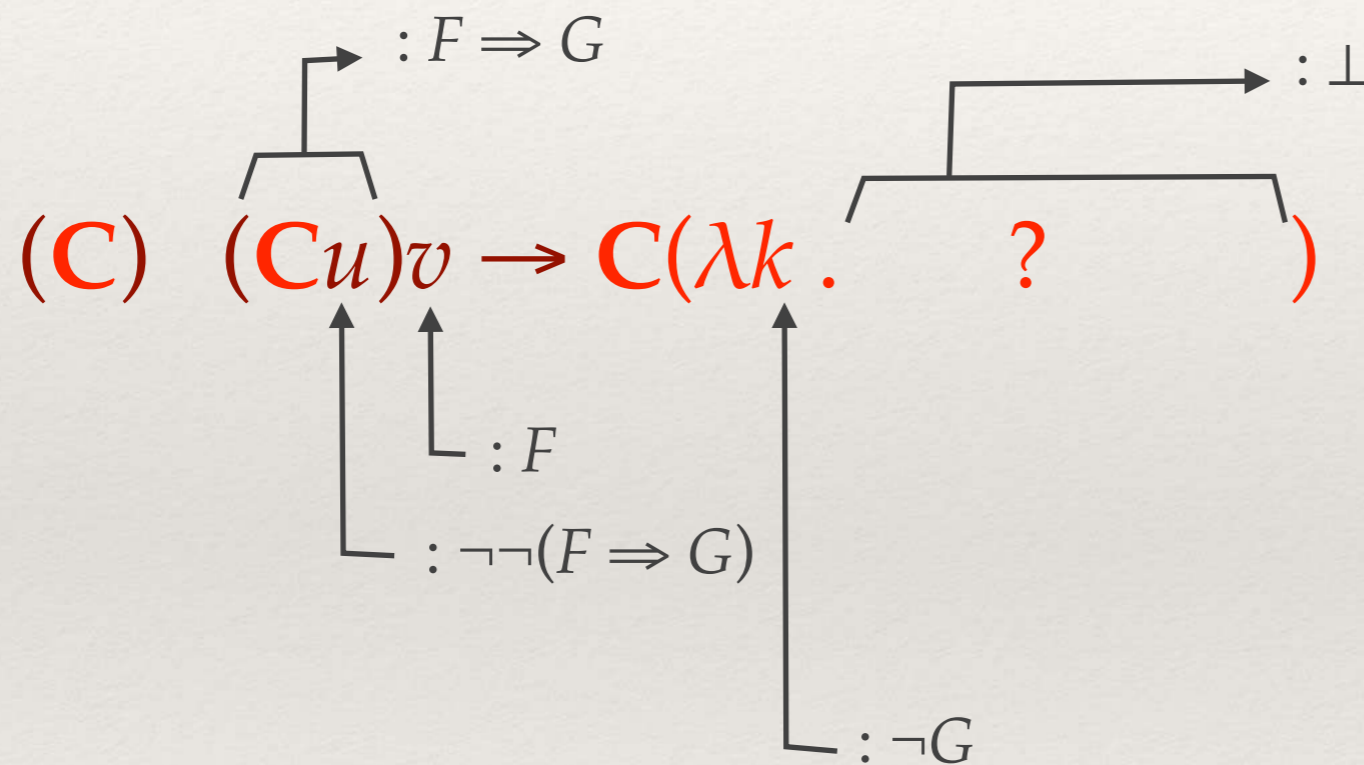
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

❖ Introduisons une abstraction

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$

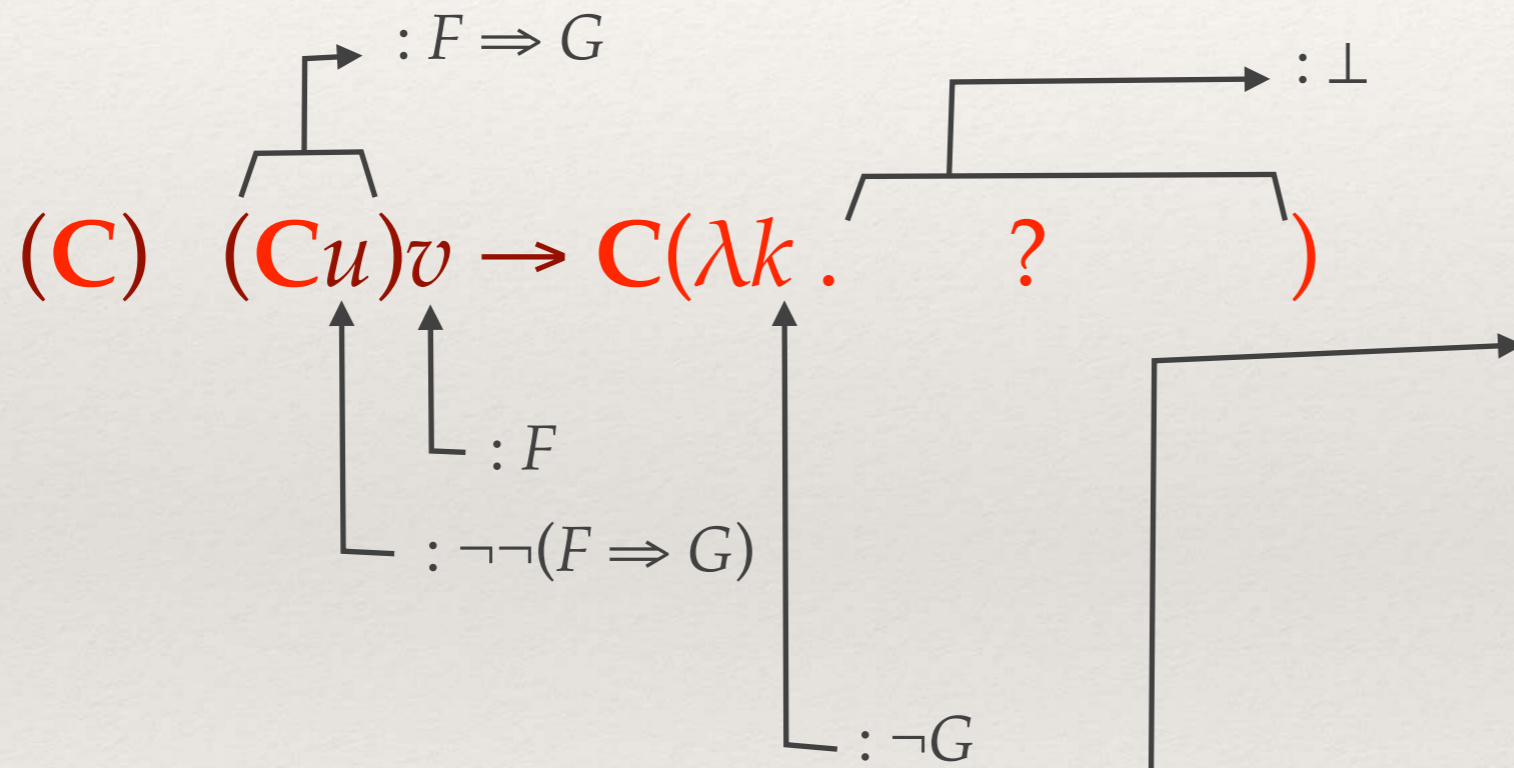


❖ Ici le terme inconnu peut être $k(qqch)$ ou $u(qqch)$
 rappel: $\neg\text{truc} = \text{truc} \Rightarrow \perp$,
 et on veut un résultat de type \perp

La règle $(Cu)v \rightarrow \dots$

- ❖ Introduisons une abstraction

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$

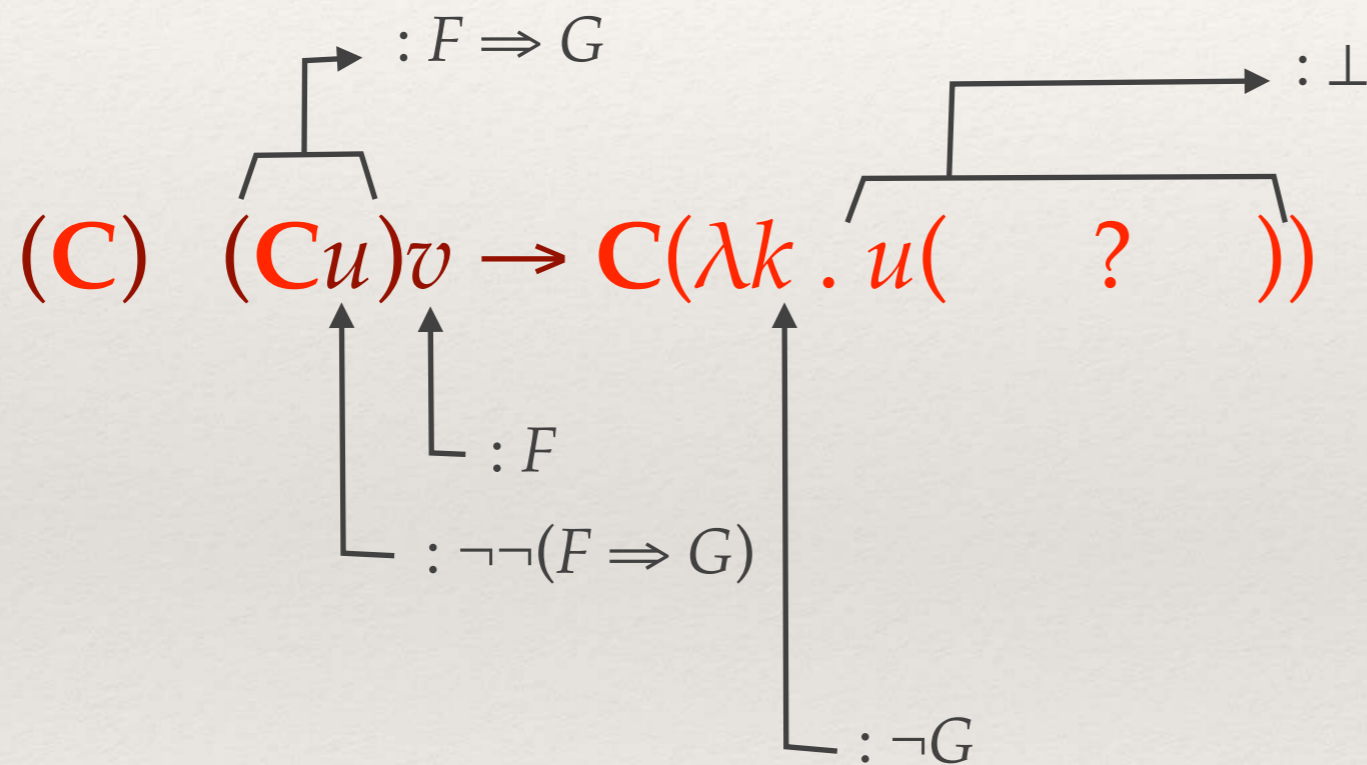


ça, ça ne marchera pas:
il faudrait trouver un qqch de type G ,
et c'est justement ce qu'on cherche depuis le début

- ❖ Ici le terme inconnu peut être $k(\text{qqch})$ ou $u(\text{qqch})$
 rappel: $\neg\text{truc} = \text{truc} \Rightarrow \perp$,
 et on veut un résultat de type \perp

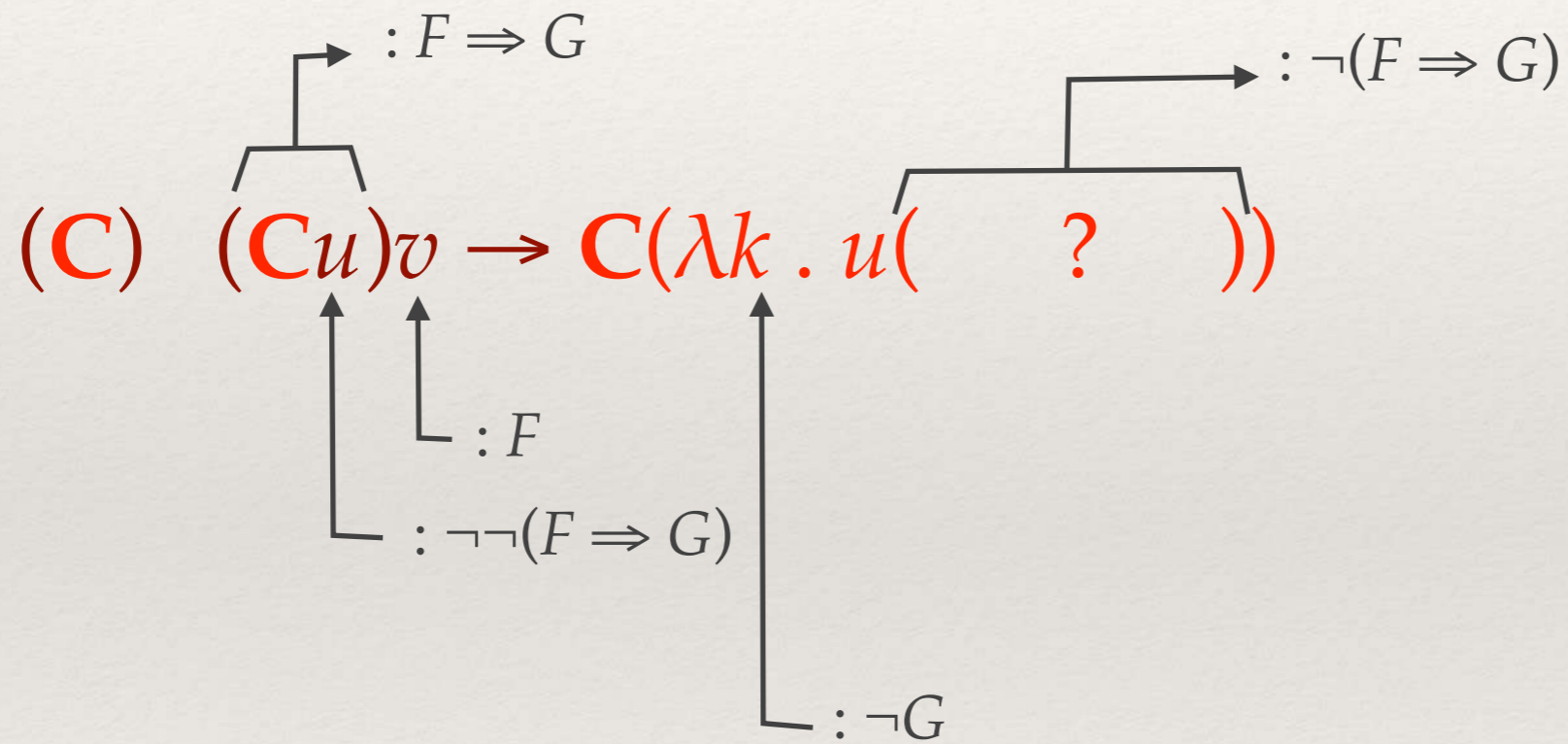
La règle $(Cu)v \rightarrow \dots$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

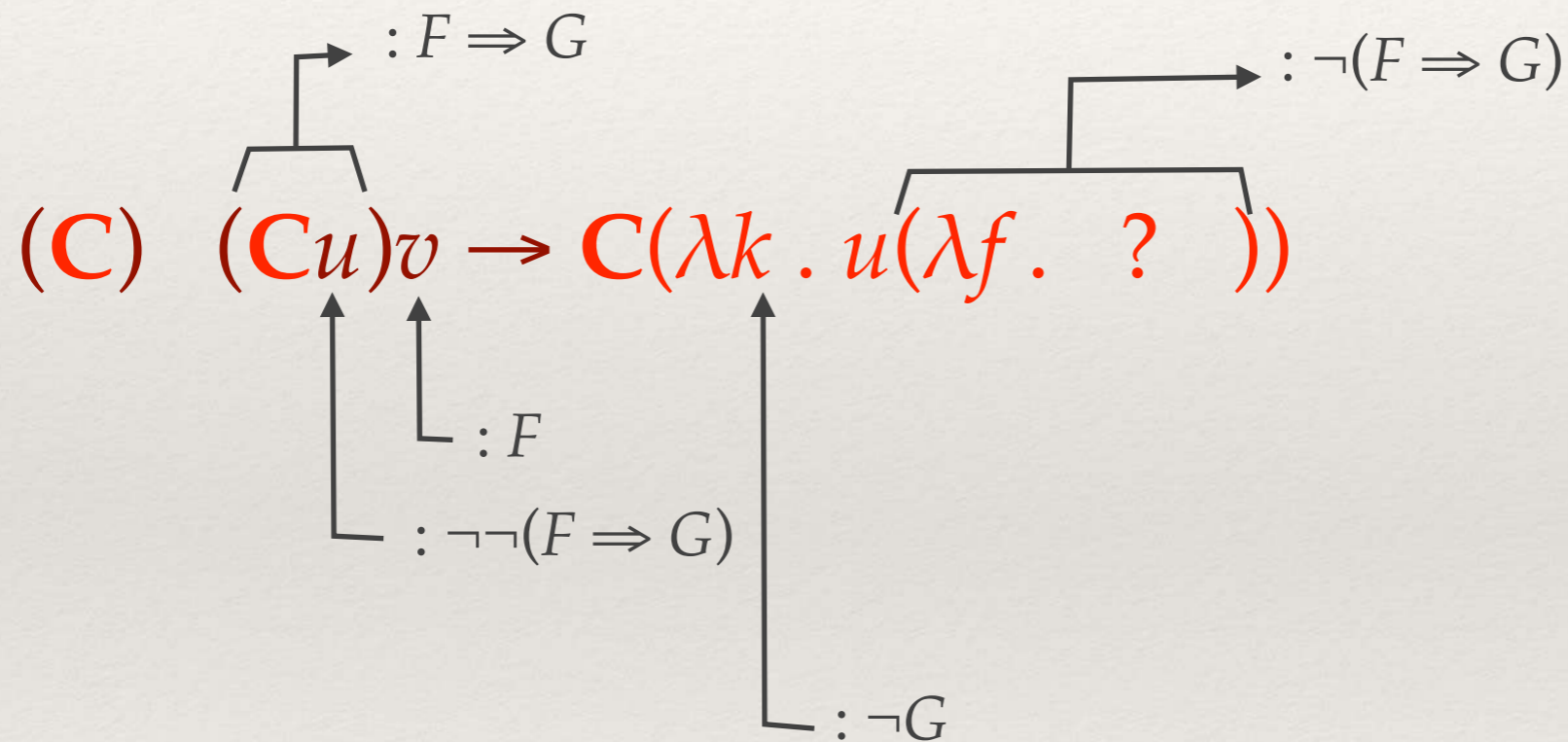
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

❖ Introduisons une abstraction

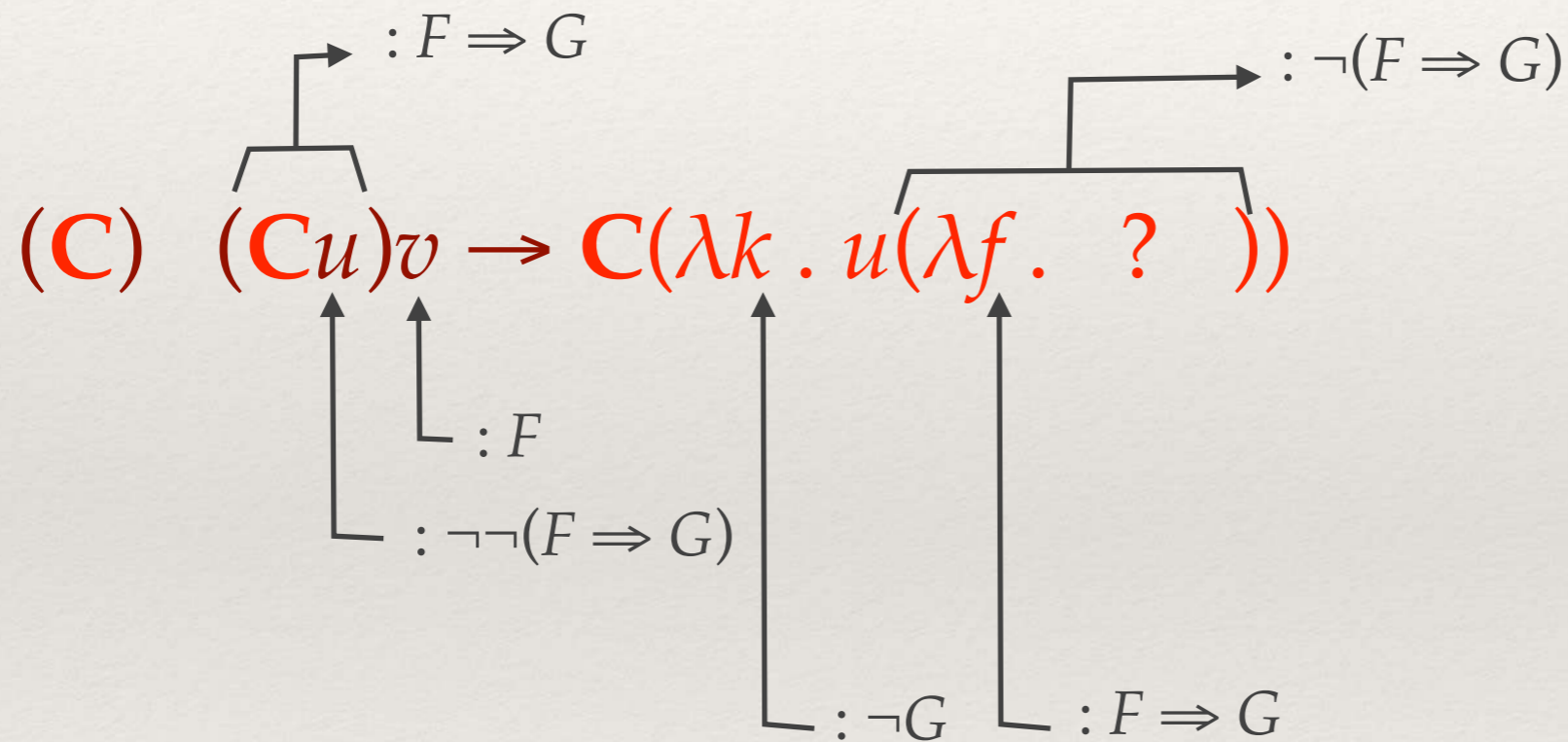
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

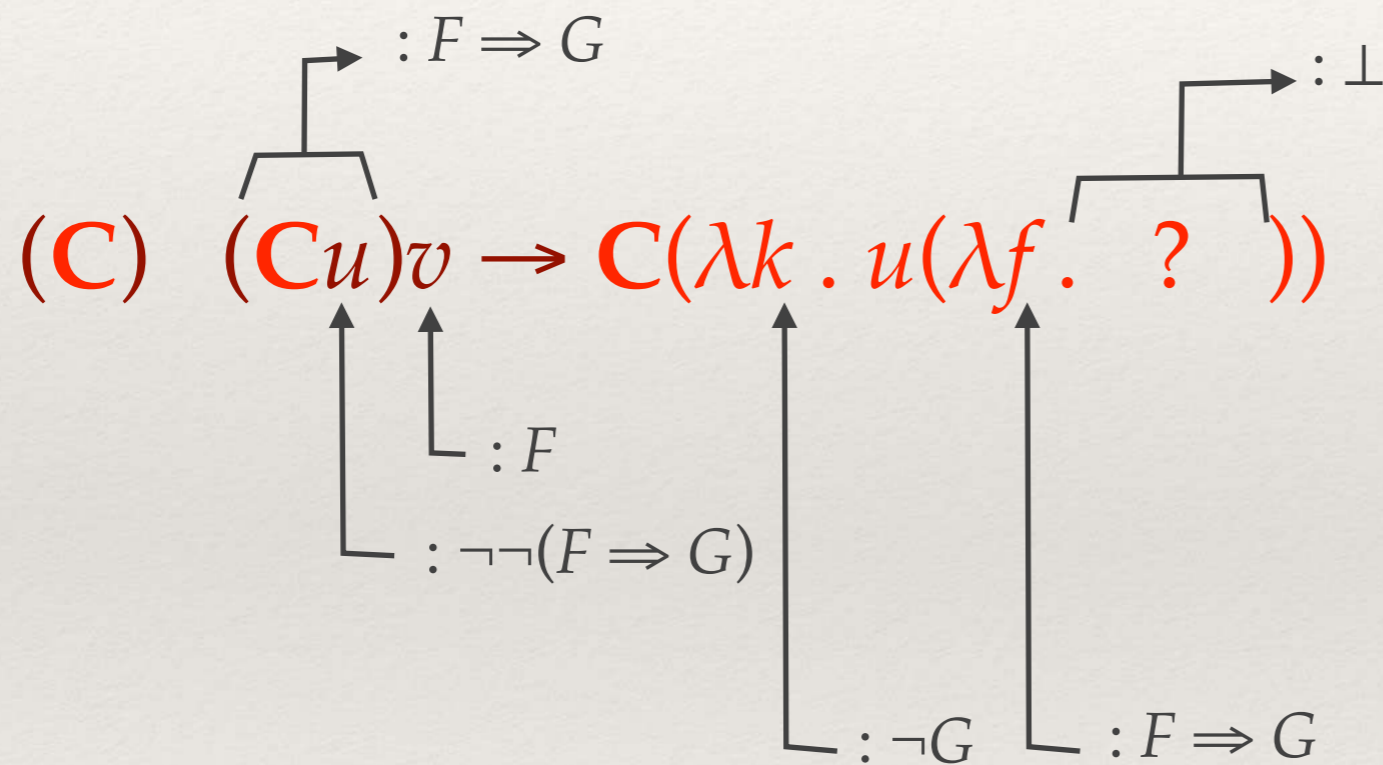
❖ Introduisons une abstraction

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



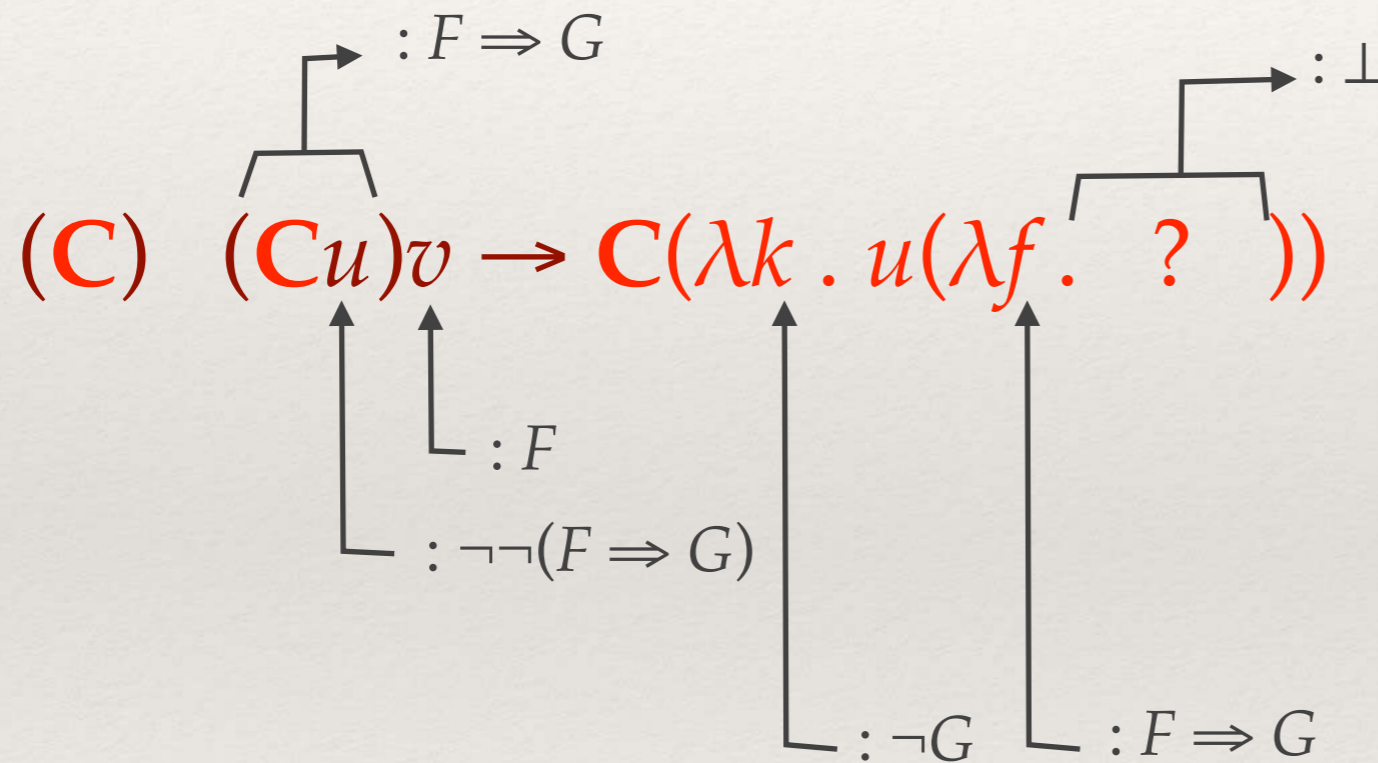
La règle $(Cu)v \rightarrow \dots$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

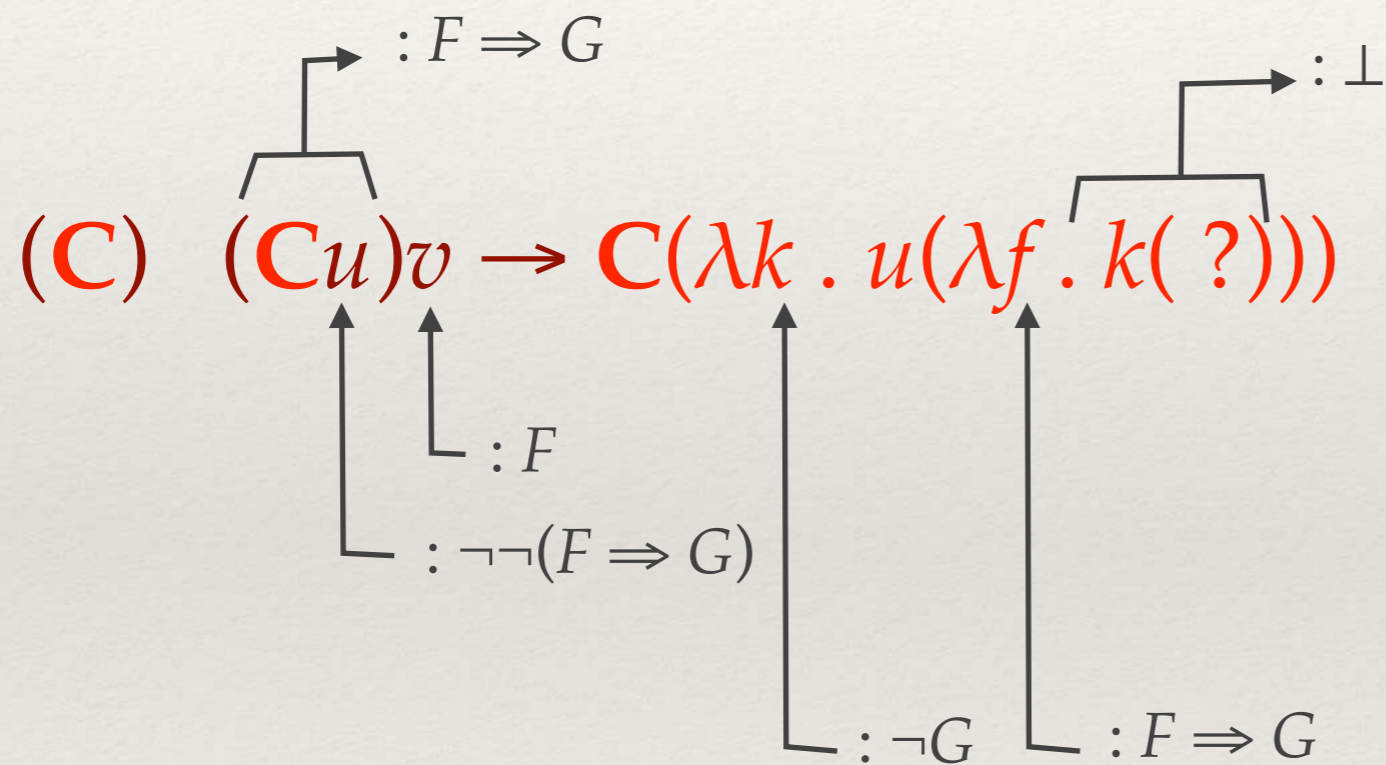
$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



- ❖ Ici le terme inconnu peut être $k(qqch)$ par ex., non?
 rappel: $\neg\text{truc} = \text{truc} \Rightarrow \perp$,
 et on veut un résultat de type \perp

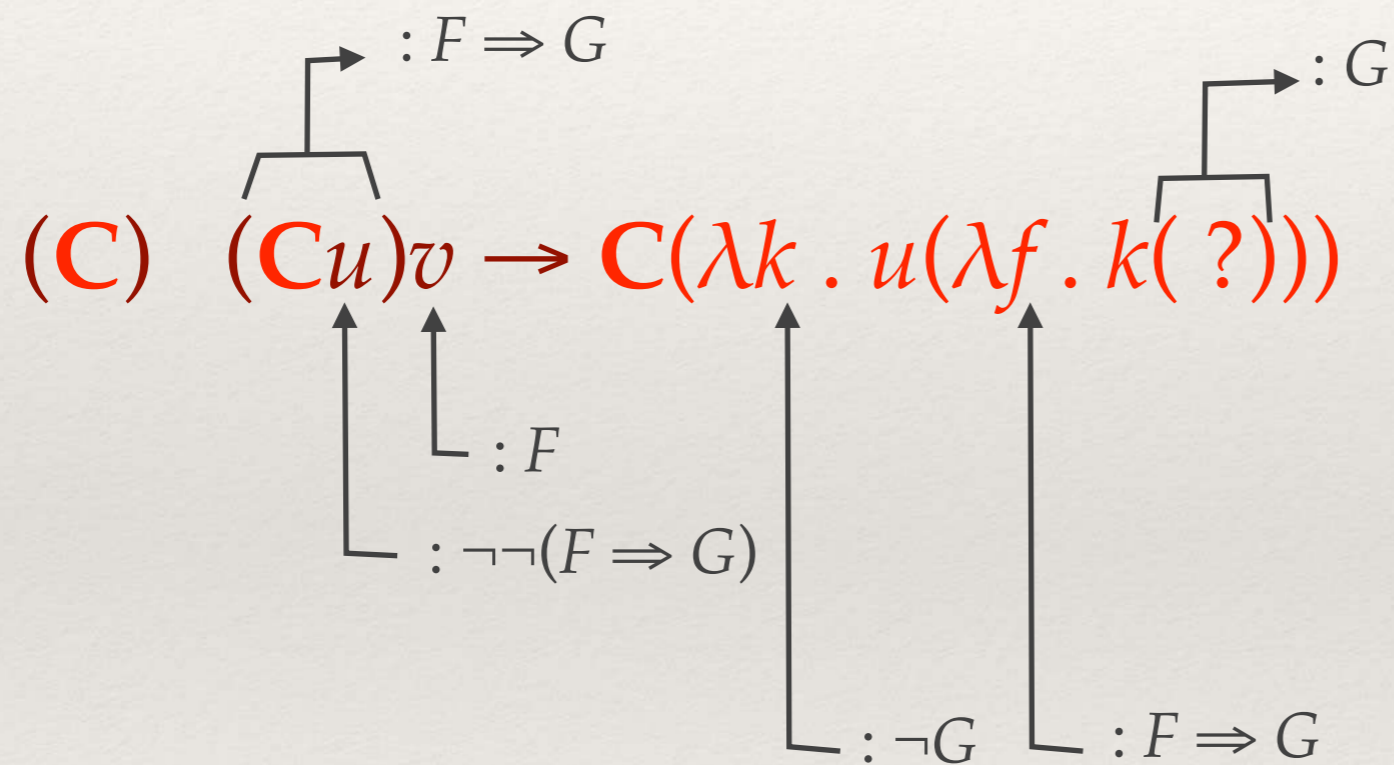
La règle $(Cu)v \rightarrow \dots$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



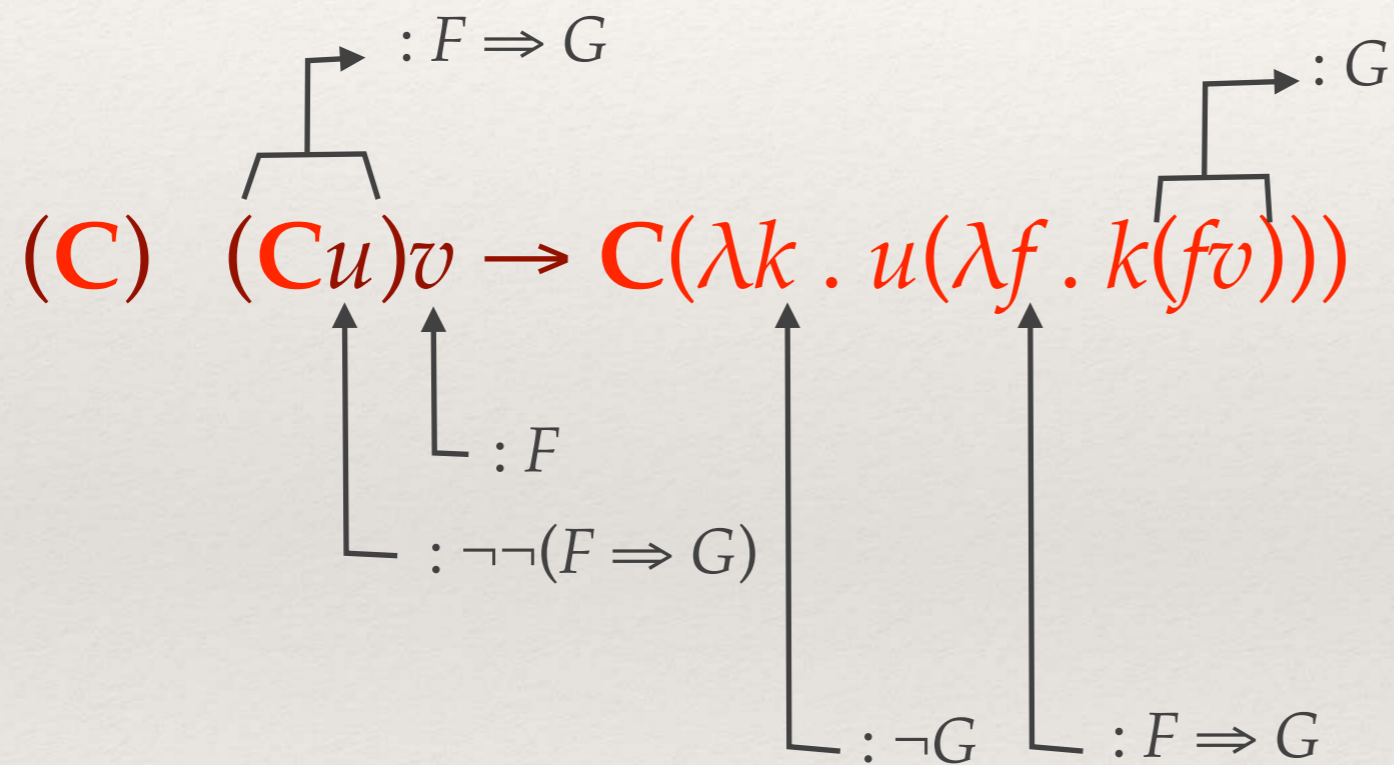
La règle $(Cu)v \rightarrow \dots$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



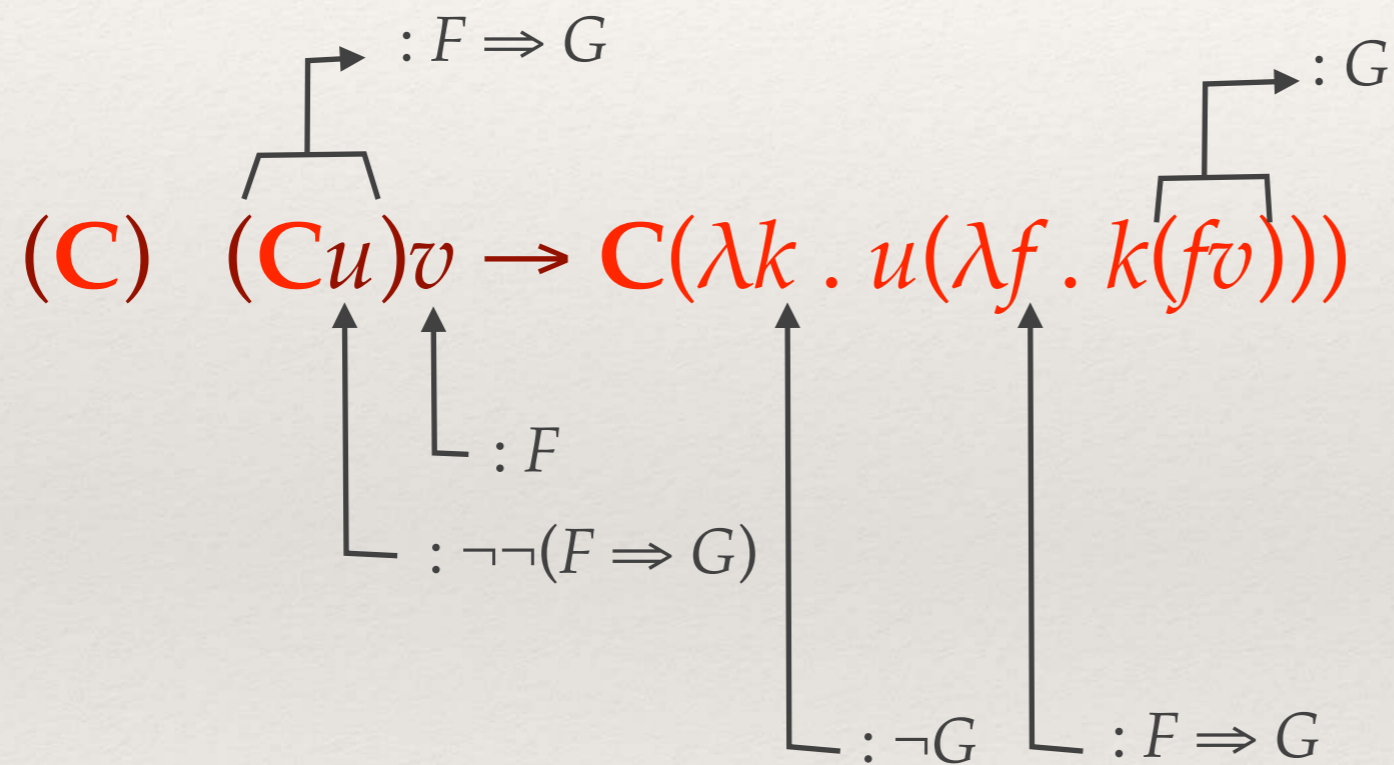
La règle $(Cu)v \rightarrow \dots$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



La règle $(Cu)v \rightarrow \dots$

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} \text{ } (\neg\neg E)$$



❖ Voilà, on a trouvé!

Logique classique

Le λ_C -calcul

❖ Types (formules):

$F, G, \dots ::= A$

| $F \Rightarrow G$

| \perp (« faux »)

❖ Termes:

$u, v, \dots ::= x$

| uv

| $\lambda x.u$

| Cu

❖ Réduction:

(β) $(\lambda x.u)v \rightarrow u[x:=v]$

(C) $(Cu)v \rightarrow$

$C(\lambda k . u(\lambda f . k(fv)))$

!

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

$$\frac{}{\Gamma, x:F \vdash x : F} (Ax)$$

$$\frac{\Gamma \vdash u : F \Rightarrow G \quad \Gamma \vdash v : F}{\Gamma \vdash uv : G} (\Rightarrow E)$$

$$\frac{\Gamma, x:F \vdash u : G}{\Gamma \vdash \lambda x.u : F \Rightarrow G} (\Rightarrow I)$$

La η -règle pour \mathbf{C}

- ❖ Dans le poly, en plus de $(\mathbf{C}) \quad (\mathbf{C}u)v \rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv)))$
j'ai ajouté $(\eta\mathbf{C}) \quad \mathbf{C}(\lambda k . ku) \rightarrow u$
 $(k \text{ non libre dans } u)$
- ❖ Je vais me concentrer sur la première
par souci de simplicité.

L'élimination du faux

❖ Avant, nous avions la règle:

$$\frac{\Gamma \vdash u : \perp}{\Gamma \vdash \nabla u : G} (\perp E)$$

L'élimination du faux

- ❖ Avant, nous avions la règle:
- ❖ Elle n'est plus nécessaire.

$$\frac{\Gamma \vdash u : \perp}{\Gamma \vdash \forall u : G} (\perp E)$$

$$\frac{\begin{array}{c} \Gamma \vdash u : \perp \\ \downarrow \text{(affaiblissement)} \\ \Gamma, k : \neg G \vdash u : \perp \end{array}}{\Gamma \vdash \lambda k. u : \neg \neg G} (\Rightarrow I)$$
$$\frac{\Gamma \vdash \lambda k. u : \neg \neg G}{\Gamma \vdash C(\lambda k. u) : G} (\neg \neg E)$$

L'élimination du faux

- ❖ Avant, nous avions la règle:
- ❖ Elle n'est plus nécessaire.

$$\frac{\Gamma \vdash u : \perp}{\Gamma \vdash \forall u : G} (\perp E)$$

- ❖ On peut définir

$$\forall u = \mathbf{C}(\lambda k.u)$$

(k variable fraîche)

$$\frac{\Gamma \vdash u : \perp}{\Gamma, k : \neg G \vdash u : \perp} \downarrow \text{(affaiblissement)}$$
$$\frac{\Gamma, k : \neg G \vdash u : \perp}{\Gamma \vdash \lambda k.u : \neg \neg G} (\Rightarrow I)$$
$$\frac{\Gamma \vdash \lambda k.u : \neg \neg G}{\Gamma \vdash \mathbf{C}(\lambda k.u) : G} (\neg \neg E)$$

L'élimination du faux

- ❖ On peut définir

$$\nabla u = \mathbf{C}(\lambda k . u)$$

(k variable fraîche)

$$\begin{aligned} (\mathbf{C}) \quad (\mathbf{C}u)v &\rightarrow \\ &\mathbf{C}(\lambda k . u(\lambda f . k(fv))) \end{aligned}$$

- ❖ et même la réduction est la bonne:

$$(\nabla u)v = \mathbf{C}(\lambda k' . u) v$$

$$\rightarrow \mathbf{C}(\lambda k . (\lambda k' . u) (\lambda f . k(fv))) \quad (\mathbf{C})$$

$$\rightarrow \mathbf{C}(\lambda k . u) \quad (\beta) \text{ (} k' \text{ fraîche!)}$$

$$= \nabla u$$

Une parenthèse historique

La découverte de l'opérateur C

- ❖ Pendant longtemps, on a cru qu'il n'y avait pas de λ -calcul adapté à la logique classique:

Lemme de Joyal. Toute catégorie cartésienne close
(\sim modèle catégorique de la logique intuitionniste)
munie d'un objet dualisant
(\sim le morphisme $F \rightarrow \neg\neg F$ est un isomorphisme)
est un préordre (\sim **toutes les preuves sont convertibles**).

La découverte de l'opérateur C

- ❖ Pendant longtemps, on a cru qu'il n'y avait pas de λ -calcul adapté à la logique classique:
- ❖ En 1988, Timothy G. Griffin s'aperçoit que l'opérateur C de Felleisen répond à la question (\pm celle que nous étudions)

Lemme de Joyal. Toute catégorie cartésienne close
(\sim modèle catégorique de la logique intuitionniste)
munie d'un objet dualisant
(\sim le morphisme $F \rightarrow \neg\neg F$ est un isomorphisme)
est un préordre (\sim toutes les preuves sont convertibles).

On a Formal Correspondence Between λ -C-Terms and Classical Proofs

— Extended Abstract —

Timothy G. Griffin
Department of Computer Science
Rice University
Houston, TX 77251-1892

October 28, 1988

1 Introduction

The propositions-as-types correspondence [How80] relates proofs in *constructive* logic to functional programs. The correspondence has intrigued those interested in the formal verification of programs with the possibility of developing programs from proofs of their specifications [BC85, Con86, Moh86]. However, one drawback of this approach to program development is that the programming languages involved are purely functional. These



La découverte de l'opérateur C

❖ Pendant longtemps, on a cru qu'il n'y avait pas de λ -calcul adapté à la logique classique:

Oups!

❖ En ~~1988~~, Timothy G. Griffin s'aperçoit que l'opérateur C de Felleisen répond

à la question (± millions)
Ben oui, on ne va pas accepter un papier qui résout un problème qu'on croyait impossible...

Lemme de Joyal. Toute catégorie cartésienne close (~modèle catégorique de la logique intuitionniste) munie d'un objet dualisant (~ le morphisme $F \rightarrow \neg\neg F$ est un isomorphisme) est un préordre (~ toutes les preuves sont convertibles).



On a Formal Correspondence Between

December 20, 1988

Dear Author:

I am sorry to have to tell you that your paper

On a Formal Correspondence Between A-C-Terms + Classical Proofs
was not among those selected by the program committee for presentation in June. Far more good papers were submitted than the program could possibly accommodate and we were forced to reject papers that were clearly publishable in a journal.

Thank you anyhow for submitting your paper to LICS 89 and I hope that you will attend the meeting in June.

Yours sincerely,

R. Parikh

Rohit Parikh
Program chair

La découverte de l'opérateur C

- ❖ Pendant longtemps, on a cru qu'il n'y avait pas de λ -calcul adapté à la logique classique:
- ❖ En ~~1988~~¹⁹⁹⁰, Timothy G. Griffin s'aperçoit que l'opérateur C de Felleisen répond à la question (\pm celle que nous étudions)

Lemme de Joyal. Toute catégorie cartésienne close
(\sim modèle catégorique de la logique intuitionniste)
munie d'un objet dualisant
(\sim le morphisme $F \rightarrow \neg\neg F$ est un isomorphisme)
est un préordre (\sim toutes les preuves sont convertibles).

Ouf, publié!



A Formulae-as-Types Notion of Control

Timothy G. Griffin*
Department of Computer Science
Rice University
Houston, TX 77251-1892

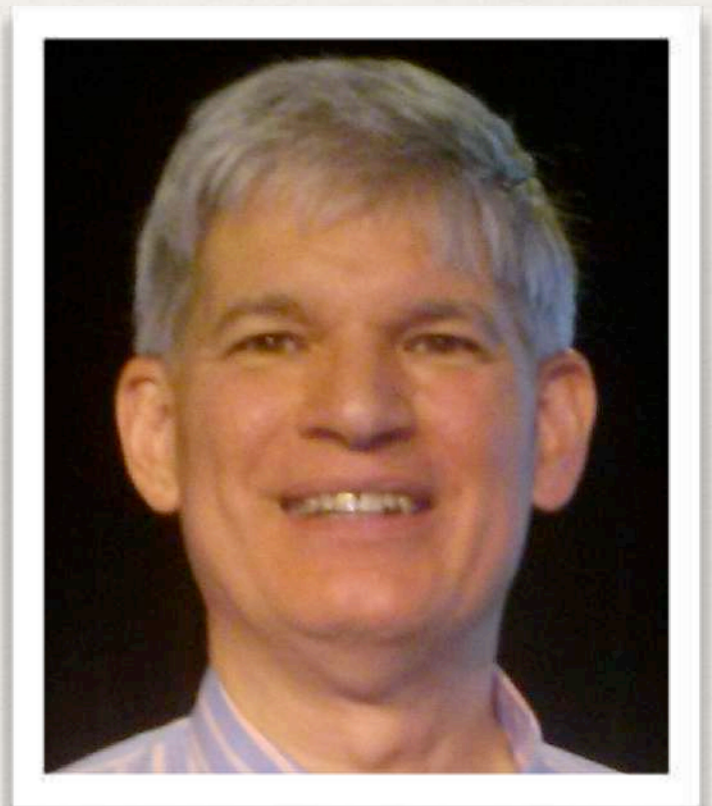
Matthias Felleisen

- ❖ Ce n'est pas T. Griffin qui a inventé l'opérateur **C**!
- ❖ C'est M. Felleisen, qui a passé des années à l'étudier, pour formaliser l'opérateur `call/cc` de Scheme...



Scheme

- ❖ Scheme est un dialecte Lisp à portée lexicale (donc en clair un λ -calcul étendu)
- ❖ inventé par Gerald P. Sussman et
Guy L. Steele Jr.
dans les années 1970
- ❖ qui contenait une primitive mystérieuse `call-with-current-continuation` (`call/cc` pour les intimes)
GLS l'a inventée comme suit...

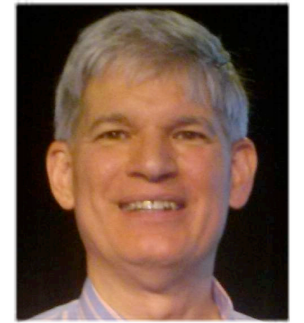


let/cc

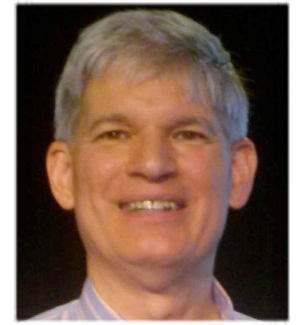
- ❖ GLS dit: imaginons que je puisse écrire

`let/cc k u`

pour dire: exécuter u après avoir lié localement
la variable k à la **continuation courante**



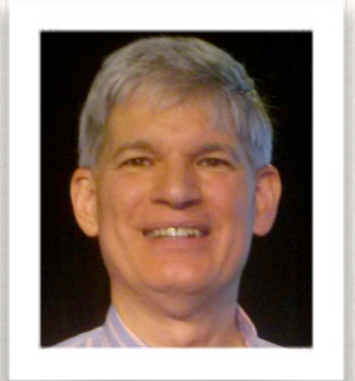
let/cc



- ❖ GLS dit: imaginons que je puisse écrire
$$\text{let/cc } k \ u$$
pour dire: exécuter u après avoir lié localement la variable k à la **continuation courante**
- ❖ ... = le contexte d'exécution courant E

(un terme à un trou, noté entre crochets)

let/cc



- ❖ GLS dit: imaginons que je puisse écrire

$\text{let/cc } k \ u$

pour dire: exécuter u après avoir lié localement
la variable k à la **continuation courante**

- ❖ ... = le contexte d'exécution courant E

(un terme à un trou, noté entre crochets)

- ❖ let/cc le **réifie** en une nouvelle constante c_E :

$E[\text{let/cc } k \ u] \rightarrow E[u[k:=c_E]]$

(règle exécutée au sommet du terme uniquement)

let/cc

- ❖ $E[\text{let/cc } k \ u] \rightarrow E[u[k:=c_E]]$
(règle exécutée au sommet du terme uniquement)
- ❖ ... et c_E fonctionne comme une **exception**
qu'on peut **lancer** par **throw**:
 $E'[\text{throw } c_E \ a] \rightarrow E[a]$
(on oublie le contexte d'exécution courant E' , et on réinstalle E)

Example

$E[\text{let/cc } k (+ 1 (\text{throw } k 3))]$

$E[\text{let/cc } k u] \rightarrow E[u[k:=c_E]]$
 $E'[\text{throw } c_E a] \rightarrow E[a]$

Example

$E[\text{let/cc } k (+ 1 (\text{throw } k 3))]$

$\rightarrow E[(+ 1 (\text{throw } c_E 3))]$

$E[\text{let/cc } k u] \rightarrow E[u[k:=c_E]]$

$E'[\text{throw } c_E a] \rightarrow E[a]$

Exemple

$E[\text{let/cc } k (+ 1 (\text{throw } k 3))]$

$\rightarrow E[(+ 1 (\text{throw } c_E 3))]$

$= E'[\text{throw } c_E 3]$

où $E' = E[(+ 1 [])]$

$E[\text{let/cc } k u] \rightarrow E[u[k:=c_E]]$
 $E'[\text{throw } c_E a] \rightarrow E[a]$

Exemple

$E[\text{let/cc } k (+ 1 (\text{throw } k 3))]$

$\rightarrow E[(+ 1 (\text{throw } c_E 3))]$

$= E'[\text{throw } c_E 3]$

où $E' = E[(+ 1 [])]$

$\rightarrow E[3]$

$E[\text{let/cc } k u] \rightarrow E[u[k:=c_E]]$
 $E'[\text{throw } c_E a] \rightarrow E[a]$

Exemple

$E[\text{let/cc } k (+ 1 (\text{throw } k 3))]$

$\rightarrow E[(+ 1 (\text{throw } c_E 3))]$

$= E'[\text{throw } c_E 3]$ où $E' = E[(+ 1 [])]$

$\rightarrow E[3]$

... autrement dit, l'exception k est lancée avec argument 3 avant qu'on aie la chance d'ajouter 1

$E[\text{let/cc } k u] \rightarrow E[u[k:=c_E]]$
 $E'[\text{throw } c_E a] \rightarrow E[a]$

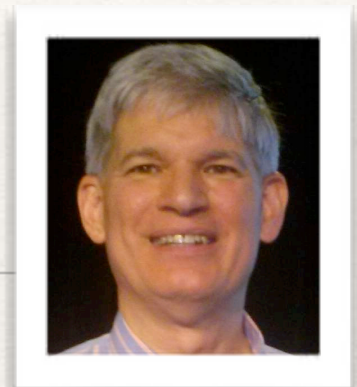
Simplification (1/3)



- ❖ GLS: plutôt que de créer une nouvelle constante c_E , on peut juste former le terme $\lambda x . E[x]$

$E[\text{let/cc } k \ u] \rightarrow E[u[k:=c_E]]$
 $E'[\text{throw } c_E \ a] \rightarrow E[a]$

Simplification (1/3)



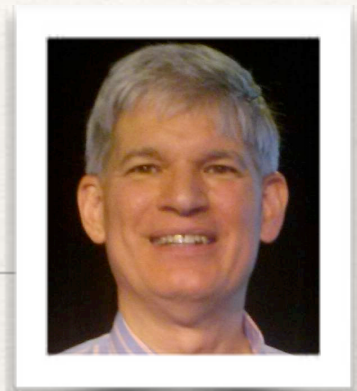
- ❖ GLS: plutôt que de créer une nouvelle constante c_E , on peut juste former le terme $\lambda x . E[x]$

$$E[\text{let/cc } k \ u] \rightarrow E[u[k:=c_E]]$$
$$E'[\text{throw } c_E \ a] \rightarrow E[a]$$

- ❖ mais alors, on peut changer les règles en:

$$E[\text{let/cc } k \ u] \rightarrow E[u[k:=\lambda x . E[x]]]$$
$$E'[\text{throw } u \ a] \rightarrow ua$$

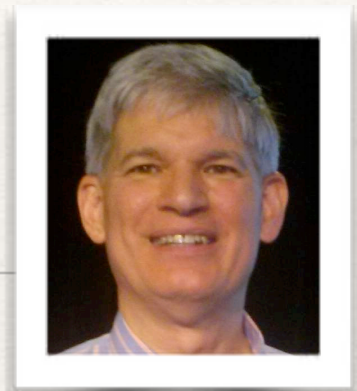
Simplification (2/3)



- ❖ GLS: `let/cc` lie la variable k , mais il serait plus élégant d'utiliser λ pour ça:
on réécrit `let/cc k u`
en `call/cc ($\lambda k . u$)`

$$E[\text{let/cc } k \ u] \rightarrow E[u[k := \lambda x . E[x]]]$$
$$E'[\text{throw } u \ a] \rightarrow ua$$

Simplification (2/3)



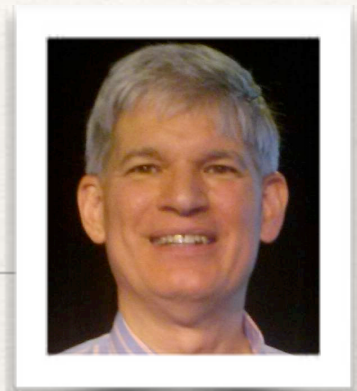
- ❖ GLS: `let/cc` lie la variable k , mais il serait plus élégant d'utiliser λ pour ça:
on réécrit `let/cc k u`
en `call/cc ($\lambda k . u$)`

$$E[\text{let/cc } k \ u] \rightarrow E[u[k := \lambda x . E[x]]]$$
$$E'[\text{throw } u \ a] \rightarrow ua$$

- ❖ Les règles deviennent:

$$E[\text{call/cc } (\lambda k . u)] \rightarrow E[u[k := \lambda x . E[x]]]$$
$$E'[\text{throw } u \ a] \rightarrow ua$$

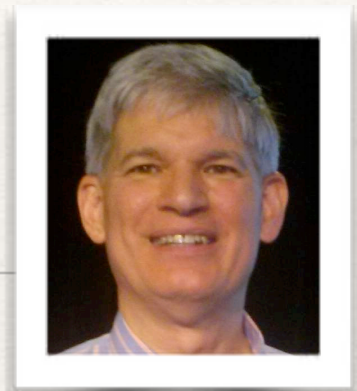
Simplification (3/3)



- ❖ GLS: comme `call/cc` est maintenant une fonction, on peut l'appliquer à un terme quelconque, pas juste une abstraction

$$E[\text{call/cc } (\lambda k . u)] \rightarrow E[u[k := \lambda x . E[x]]]$$
$$E'[\text{throw } u \ a] \rightarrow ua$$

Simplification (3/3)



- ❖ GLS: comme `call/cc` est maintenant une fonction, on peut l'appliquer à un terme quelconque, pas juste une abstraction

$$E[\text{call/cc } (\lambda k . u)] \rightarrow E[u[k := \lambda x . E[x]]]$$
$$E'[\text{throw } u a] \rightarrow ua$$

- ❖ Les règles deviennent:

$$E[\text{call/cc } u] \rightarrow E[u(\lambda x . E[x])]$$
$$E'[\text{throw } u a] \rightarrow ua$$

Un cas particulier

- ❖ En appel par nom
(réduction de tête faible),
les contextes E sont des compositions de contextes
élémentaires $[] v$ d'application à des arguments v

Un cas particulier

- ❖ En appel par nom
(réduction de tête faible),

les contextes E sont des compositions de contextes élémentaires $[] v$ d'application à des arguments v

$$E[\text{call/cc } u] \rightarrow E[u(\lambda x . E[x])]$$

$$E'[\text{throw } u a] \rightarrow ua$$

Un cas particulier

- ❖ En appel par nom
(réduction de tête faible),
les contextes E sont des compositions de contextes
élémentaires $[] v$ d'application à des arguments v

$$E[\text{call/cc } u] \rightarrow E[u(\lambda x . E[x])]$$
$$E'[\text{throw } u a] \rightarrow ua$$

- ❖ Felleisen décompose la première règle en:

$$(\text{call/cc } u) v \rightarrow \text{call/cc}(\lambda k.u(\lambda f.k(fv)))v$$

($E=[] v$, x est renommée en f ; on garde un $\text{call/cc}(\lambda k\dots)$

à droite qui mangera le reste du contexte)

plus une règle qui effacera le call/cc au sommet du terme

(je ne décrirai pas toutes les étapes par lesquelles il passe...)

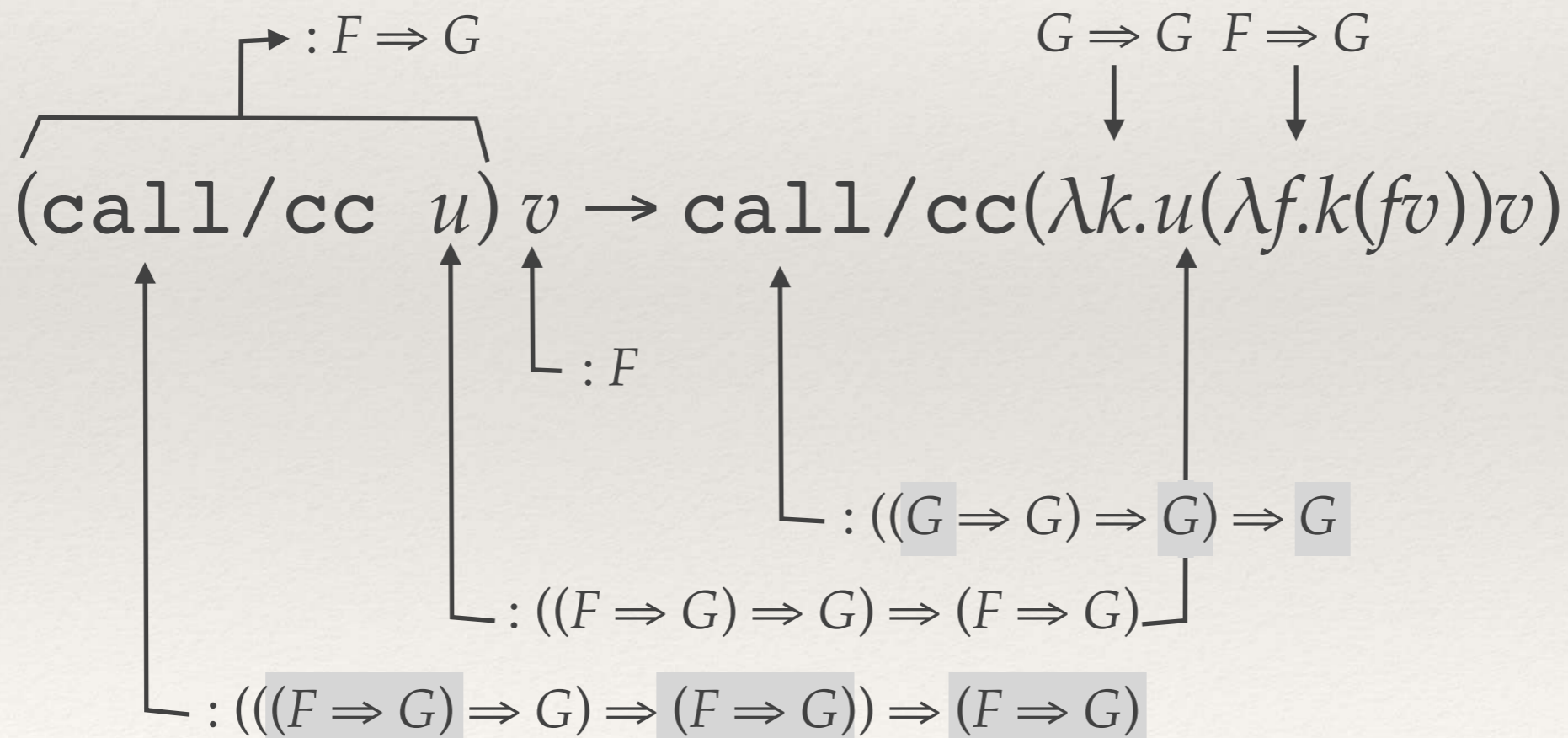


On peut typer call/cc!

- ❖ Ce que T. Griffin a découvert, en réalité:

$$\text{call/cc} : ((F \Rightarrow G) \Rightarrow F) \Rightarrow F$$

(loi de Peirce)



$$\text{call/cc} + \nabla \cong \mathbf{C}$$

$$\text{call/cc} : ((F \Rightarrow G) \Rightarrow F) \Rightarrow F$$

$$\mathbf{C} : \neg\neg F \Rightarrow F$$

- ❖ \mathbf{C} est une version simplifiée de $\text{call/cc} + \nabla$ due à Felleisen (encore):

$$\text{call/cc} + \nabla \cong \mathbf{C}$$

$$\text{call/cc} : ((F \Rightarrow G) \Rightarrow F) \Rightarrow F$$

$$\mathbf{C} : \neg\neg F \Rightarrow F$$

- ❖ \mathbf{C} est une version simplifiée de $\text{call/cc} + \nabla$ due à Felleisen (encore):
- ❖ (\leftarrow) On peut définir $\mathbf{C}u$ (pour $u:\neg\neg F$) comme $\text{call/cc}(\lambda k_{\neg F} . \nabla(uk))$

call/cc + $\nabla \cong \mathbf{C}$

$$\text{call/cc} : ((F \Rightarrow G) \Rightarrow F) \Rightarrow F$$

$$\mathbf{C} : \neg\neg F \Rightarrow F$$

- ❖ \mathbf{C} est une version simplifiée de $\text{call/cc} + \nabla$ due à Felleisen (encore):
- ❖ (\leftarrow) On peut définir $\mathbf{C}u$ (pour $u:\neg\neg F$) comme
$$\text{call/cc}(\lambda k_{\neg F} . \nabla(uk))$$
- ❖ (\rightarrow) On peut définir call/cc comme
$$\lambda z_{(F \Rightarrow G) \Rightarrow F} . \mathbf{C}(\lambda k_{\neg F} . k(z(\lambda x_F . \nabla(kx))))$$
 (et ∇u comme $\mathbf{C}(\lambda k.u)$)

(Je vous laisse vérifier le typage; et les réductions — ça, je ne l'ai pas fait...)

Les autres connecteurs logiques en logique classique

Le « et »

- ❖ J'ai dit qu'on pouvait définir $F \wedge G = \neg(F \Rightarrow \neg G)$
c'est-à-dire $F \wedge G = (F \Rightarrow G \Rightarrow \perp) \Rightarrow \perp$

Le « et »

- ❖ J'ai dit qu'on pouvait définir $F \wedge G = \neg(F \Rightarrow \neg G)$
c'est-à-dire $F \wedge G = (F \Rightarrow G \Rightarrow \perp) \Rightarrow \perp$
- ❖ Le codage s'étend aux λ -termes:
 - $\langle u, v \rangle = \lambda f_{F \Rightarrow G \Rightarrow \perp} . f u v$ (comme en λ -calcul non typé)
 - $\pi_1 u = \mathbf{C}(\lambda k_{\neg F} . u(\lambda x_F . \lambda y_G . k x))$ (... ou presque)
 - $\pi_2 u = \mathbf{C}(\lambda k_{\neg F} . u(\lambda x_F . \lambda y_G . k y))$

Le « et »

- ❖ J'ai dit qu'on pouvait définir $F \wedge G = \neg(F \Rightarrow \neg G)$
c'est-à-dire $F \wedge G = (F \Rightarrow G \Rightarrow \perp) \Rightarrow \perp$

- ❖ Le codage s'étend aux λ -termes:

$$\langle u, v \rangle = \lambda f_{F \Rightarrow G \Rightarrow \perp} . fuv \quad (\text{comme en } \lambda\text{-calcul non typé})$$

$$\pi_1 u = \mathbf{C}(\lambda k_{\neg F} . u(\lambda x_F . \lambda y_G . kx)) \quad (\dots \text{ ou presque})$$

$$\pi_2 u = \mathbf{C}(\lambda k_{\neg F} . u(\lambda x_F . \lambda y_G . ky))$$

$$\begin{aligned} (\mathbf{C}) \quad (\mathbf{C}u)v &\rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv))) \\ (\eta\mathbf{C}) \quad \mathbf{C}(\lambda k . ku) &\rightarrow u \end{aligned}$$

- ❖ et aux réductions!

$$\pi_1 \langle u, v \rangle = \mathbf{C}(\lambda k_{\neg F} . (\lambda f . fuv)(\lambda x . \lambda y . kx))$$

$$\rightarrow \mathbf{C}(\lambda k_{\neg F} . (\lambda x . \lambda y . kx)uv)$$

$$\rightarrow^2 \mathbf{C}(\lambda k_{\neg F} . ku) \rightarrow u \quad \text{par } (\eta\mathbf{C})$$

Le « ou »

- ❖ On peut définir $F \vee G = (\neg F) \Rightarrow (\neg\neg G) = (F \Rightarrow \perp) \Rightarrow (G \Rightarrow \perp) \Rightarrow \perp$
(ça fonctionne mieux que $F \vee G = (\neg F) \Rightarrow G$, que je vous avais proposé au début)

Le « ou »

- ❖ On peut définir $F \vee G = (\neg F) \Rightarrow (\neg\neg G) = (F \Rightarrow \perp) \Rightarrow (G \Rightarrow \perp) \Rightarrow \perp$
(ça fonctionne mieux que $F \vee G = (\neg F) \Rightarrow G$, que je vous avais proposé au début)

- ❖ Le codage s'étend aux λ -termes:

$$l_1 u = \lambda k_1 F \Rightarrow \perp . \lambda k_2 G \Rightarrow \perp . k_1 u \quad (\text{comme en } \lambda\text{-calcul non typé})$$

$$l_2 u = \lambda k_1 F \Rightarrow \perp . \lambda k_2 G \Rightarrow \perp . k_2 u$$

$$\text{case } u \text{ of } l_1 x_1 \mapsto v_1 \mid l_2 x_2 \mapsto v_2$$

$$= \mathbf{C}(\lambda k_{\neg H} . u(\lambda x_1 . k v_1)(\lambda x_2 . k v_2)) \quad (\dots \text{ ou presque})$$

Le « ou »

- ❖ On peut définir $F \vee G = (\neg F) \Rightarrow (\neg\neg G) = (F \Rightarrow \perp) \Rightarrow (G \Rightarrow \perp) \Rightarrow \perp$
(ça fonctionne mieux que $F \vee G = (\neg F) \Rightarrow G$, que je vous avais proposé au début)

- ❖ Le codage s'étend aux λ -termes:

$$l_1 u = \lambda k_1 F \Rightarrow \perp . \lambda k_2 G \Rightarrow \perp . k_1 u \quad (\text{comme en } \lambda\text{-calcul non typé})$$

$$l_2 u = \lambda k_1 F \Rightarrow \perp . \lambda k_2 G \Rightarrow \perp . k_2 u$$

$$\begin{aligned} \text{case } u \text{ of } l_1 x_1 \mapsto v_1 \mid l_2 x_2 \mapsto v_2 \\ = \mathbf{C}(\lambda k_{\neg H} . u(\lambda x_1 . k v_1)(\lambda x_2 . k v_2)) \end{aligned}$$

$$\begin{aligned} (\mathbf{C}) \quad (\mathbf{C}u)v \rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv))) \\ (\eta\mathbf{C}) \quad \mathbf{C}(\lambda k . ku) \rightarrow u \end{aligned}$$

- ❖ et aux réductions!

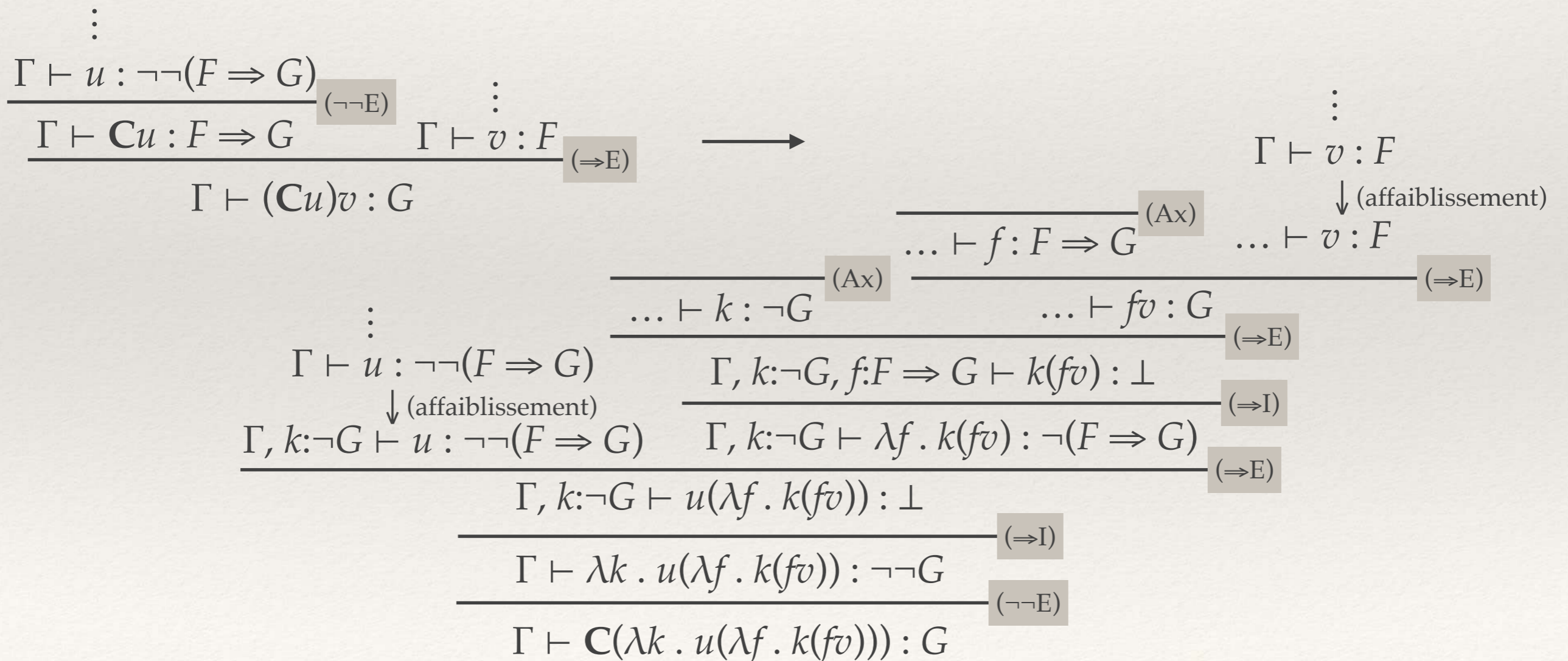
$$\begin{aligned} \text{case } l_1 u \text{ of } l_1 x_1 \mapsto v_1 \mid l_2 x_2 \mapsto v_2 &= \mathbf{C}(\lambda k . (\lambda k_1 . \lambda k_2 . k_1 u)(\lambda x_1 . k v_1)(\lambda x_2 . k v_2)) \\ &\rightarrow^2 \mathbf{C}(\lambda k . (\lambda x_1 . k v_1)u) \\ &\rightarrow \mathbf{C}(\lambda k . k(v_1[x:=u])) \rightarrow v_1[x:=u] \quad \text{par } (\eta\mathbf{C}) \end{aligned}$$

Normalisation forte

La règle (C)

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 (η C) $C(\lambda k . ku) \rightarrow u$

❖ Il peut paraître improbable qu'avec une « simplification » de preuve pareille, ça termine toujours

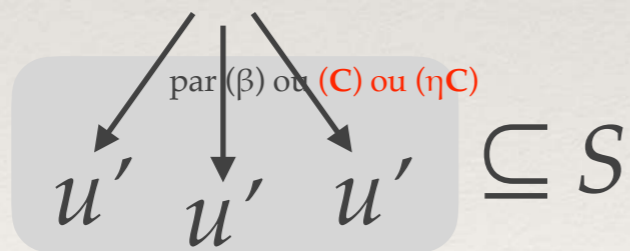


Normalisation forte

- ❖ **Thm.** Tout terme typable (dans ce nouveau système de types) est fortement normalisable.
- ❖ La démonstration est comme en logique minimale...
je vous montre juste les changements (en rouge).

Candidats de réductibilité

- ❖ Un ensemble de $\lambda\mathbf{C}$ -termes S est un **candidat de réductibilité** si et seulement si:
 - ❖ (CR1) $S \subseteq SN$ ($SN =$ les fortement normalisables pour $(\beta)+(\mathbf{C})+(\eta\mathbf{C})$)
 - ❖ (CR2) si $u \in S \xrightarrow{\text{par } (\beta) \text{ ou } (\mathbf{C}) \text{ ou } (\eta\mathbf{C})} u'$ alors $u' \in S$ (*on suit les réductions en avant*)
 - ❖ (CR3) si u neutre alors $u \in S$ (*réductions en arrière*)



où u **neutre** ssi ne commence ni par λ ni par \mathbf{C} .

Candidats de réductibilité

❖ **Lemme A.** Si S et S' sont des candidats de réductibilité, alors $S \Rightarrow S'$ aussi.

(Preuve: comme les dernières fois.)

❖ On pose $RED_A = SN$ pour toute formule atomique A ,
 $RED_{F \Rightarrow G} = RED_F \Rightarrow RED_G$,
 $RED_{\perp} = SN$

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si $\begin{array}{c} u \text{ neutre} \\ \swarrow \quad \downarrow \quad \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$ alors $u \in S$

$S \Rightarrow S' = \{u \mid \forall v \in S, uv \in S'\}$

Candidats de réductibilité

❖ **Lemme A.** Si S et S' sont des candidats de réductibilité, alors $S \Rightarrow S'$ aussi.

(Preuve: comme les dernières fois.)

❖ On pose $RED_A = SN$ pour toute formule atomique A ,

$$RED_{F \Rightarrow G} = RED_F \Rightarrow RED_G,$$

$$RED_{\perp} = SN$$

❖ SN vérifie (CR1)–(CR3) (facile). Donc:

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si $\begin{array}{c} u \text{ neutre} \\ \swarrow \quad \downarrow \quad \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$ alors $u \in S$

$$S \Rightarrow S' = \{u \mid \forall v \in S, uv \in S'\}$$

Candidats de réductibilité

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si $\begin{array}{c} u \text{ neutre} \\ \swarrow \quad \downarrow \quad \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$ alors $u \in S$

- ❖ **Lemme A.** Si S et S' sont des candidats de réductibilité, alors $S \Rightarrow S'$ aussi.

(Preuve: comme les dernières fois.)

$S \Rightarrow S' = \{u \mid \forall v \in S, uv \in S'\}$

- ❖ On pose $\text{RED}_A = \text{SN}$ pour toute formule atomique A ,
 $\text{RED}_{F \Rightarrow G} = \text{RED}_F \Rightarrow \text{RED}_G$,
 $\text{RED}_\perp = \text{SN}$

- ❖ SN vérifie (CR1)–(CR3) (facile). Donc:

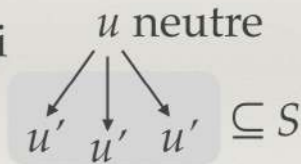
- ❖ **Lemme B.** Pour tout type F , RED_F est un candidat.

(Preuve: par récurrence sur F .)

Un lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

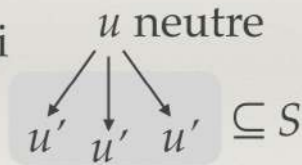
(CR3) si u neutre alors $u \in S$


- ❖ **Lemme C.** Si (pour tout $v \in \text{RED}_F$, $s[x:=v] \in \text{RED}_G$)
alors $\lambda x.s \in \text{RED}_{F \Rightarrow G}$

Un lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

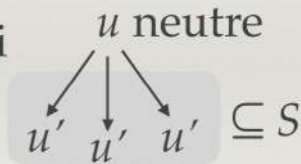
(CR3) si u neutre alors $u \in S$


- ❖ **Lemme C.** Si (pour tout $v \in \text{RED}_F$, $s[x:=v] \in \text{RED}_G$)
alors $\lambda x.s \in \text{RED}_{F \Rightarrow G}$
- ❖ Preuve: comme la dernière fois, à coups de (CR3).

Un lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

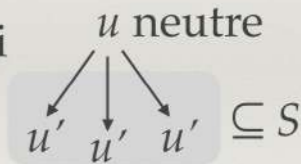
(CR3) si u neutre alors $u \in S$
 $\subseteq S$

- ❖ **Lemme C.** Si (pour tout $v \in \text{RED}_F$, $s[x:=v] \in \text{RED}_G$)
alors $\lambda x.s \in \text{RED}_{F \Rightarrow G}$
- ❖ Preuve: comme la dernière fois, à coups de (CR3).
- ❖ Permettait de traiter de la règle (\Rightarrow I)

Un lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

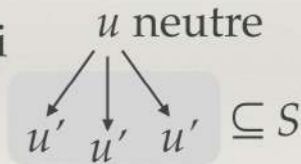
(CR3) si u neutre alors $u \in S$
 $\subseteq S$

- ❖ **Lemme C.** Si (pour tout $v \in \text{RED}_F$, $s[x:=v] \in \text{RED}_G$)
alors $\lambda x.s \in \text{RED}_{F \Rightarrow G}$
- ❖ Preuve: comme la dernière fois, à coups de (CR3).
- ❖ Permettait de traiter de la règle (\Rightarrow I)
- ❖ Nous avons une nouvelle règle ($\neg\neg$ E), pour laquelle nous démontrons le...

Un autre lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


❖ **Lemme D.** Pour tout type F , si $u \in \text{RED}_{\neg\neg F}$,
alors $\mathbf{C}u \in \text{RED}_F$

(C) $(\mathbf{C}u)v \rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv)))$

($\eta\mathbf{C}$) $\mathbf{C}(\lambda k . ku) \rightarrow u$

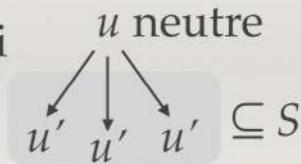
Un autre lemme utile

Preuve: par récurrence sur F

Lemme D. Pour tout type F , si $u \in \text{RED}_{\neg\neg F}$,
alors $\mathbf{C}u \in \text{RED}_F$

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


(C) $(\mathbf{C}u)v \rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv)))$

($\eta\mathbf{C}$) $\mathbf{C}(\lambda k . ku) \rightarrow u$

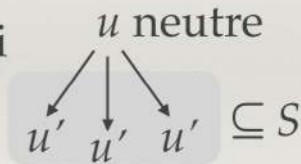
Un autre lemme utile

Preuve: par récurrence sur F

Lemme D. Pour tout type F , si $u \in \text{RED}_{\neg\neg F}$,
alors $\mathbf{C}u \in \text{RED}_F$

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


(C) $(\mathbf{C}u)v \rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv)))$

($\eta\mathbf{C}$) $\mathbf{C}(\lambda k . ku) \rightarrow u$

❖ Si F atomique ou \perp , on veut démontrer $\mathbf{C}u \in \text{SN} = \text{RED}_F$

Or toutes les réductions se passent dans u , sauf si on utilise ($\eta\mathbf{C}$).

Imaginons une réduction infinie $\mathbf{C}u \rightarrow^\infty \dots$ Deux cas:

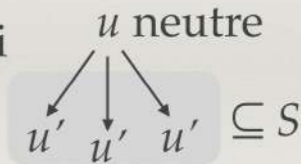
Un autre lemme utile

Preuve: par récurrence sur F

Lemme D. Pour tout type F , si $u \in \text{RED}_{\neg\neg F}$,
alors $\mathbf{C}u \in \text{RED}_F$

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


(C) $(\mathbf{C}u)v \rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv)))$

($\eta\mathbf{C}$) $\mathbf{C}(\lambda k . ku) \rightarrow u$

❖ Si F atomique ou \perp , on veut démontrer $\mathbf{C}u \in \text{SN} = \text{RED}_F$

Or toutes les réductions se passent dans u , sauf si on utilise ($\eta\mathbf{C}$).

Imaginons une réduction infinie $\mathbf{C}u \rightarrow^\infty \dots$. Deux cas:

❖ $\mathbf{C}u_0 \rightarrow \mathbf{C}u_1 \rightarrow \mathbf{C}u_2 \rightarrow \dots \rightarrow \mathbf{C}u_n \rightarrow^\infty \dots$

où $u = u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_n \rightarrow^\infty \dots$: non car $u \in \text{RED}_{\neg\neg F} \subseteq \text{SN}$

Un autre lemme utile

Preuve: par récurrence sur F

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{array}{c} u \\ \swarrow \downarrow \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$

Lemme D. Pour tout type F , si $u \in RED_{\neg\neg F}$,
 alors $Cu \in RED_F$

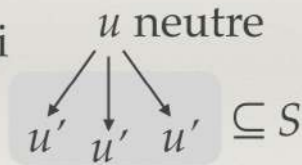
(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 (ηC) $C(\lambda k . ku) \rightarrow u$

- ❖ Si F atomique ou \perp , on veut démontrer $Cu \in SN = RED_F$
 Or toutes les réductions se passent dans u , sauf si on utilise (ηC).
 Imaginons une réduction infinie $Cu \rightarrow^\infty \dots$. Deux cas:
- ❖ $Cu_0 \rightarrow Cu_1 \rightarrow Cu_2 \rightarrow \dots \rightarrow Cu_n \rightarrow^\infty \dots$
 où $u = u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_n \rightarrow^\infty \dots$: non car $u \in RED_{\neg\neg F} \subseteq SN$
- ❖ $Cu_0 \rightarrow Cu_n = C(\lambda k . kv) \xrightarrow{(\eta C)} v \rightarrow v_1 \rightarrow v_2 \rightarrow^\infty \dots$
 où $u = u_0 \rightarrow^* u_n = \lambda k . kv$: non car sinon
 $u = u_0 \rightarrow^* u_n = \lambda k . kv \rightarrow \lambda k . kv_1 \rightarrow \lambda k . kv_2 \rightarrow^\infty \dots$

Un autre lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


❖ ... reste à démontrer que

$Cu \in \text{RED}_{F \Rightarrow G}$, sachant que

(H) $u \in \text{RED}_{\neg\neg(F \Rightarrow G)}$

(Hrec) pour tout terme $w \in \text{RED}_{\neg\neg G}$, $Cw \in \text{RED}_G$

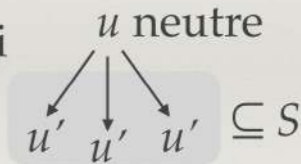
(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$

(η C) $C(\lambda k . ku) \rightarrow u$

Un autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


❖ ... reste à démontrer que

$Cu \in RED_{F \Rightarrow G}$, sachant que

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

❖ Par définition de $RED_{F \Rightarrow G}$, il suffit de démontrer que pour tout $v \in RED_F$, $(Cu)v$ est dans RED_G

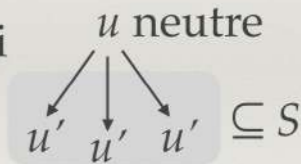
(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$

(η C) $C(\lambda k . ku) \rightarrow u$

Un autre lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


❖ ... reste à démontrer que

$\mathbf{C}u \in \text{RED}_{F \Rightarrow G}$, sachant que

(H) $u \in \text{RED}_{\neg\neg(F \Rightarrow G)}$

(Hrec) pour tout terme $w \in \text{RED}_{\neg\neg G}$, $\mathbf{C}w \in \text{RED}_G$

❖ Par définition de $\text{RED}_{F \Rightarrow G}$, il suffit de démontrer que pour tout $v \in \text{RED}_F$, $(\mathbf{C}u)v$ est dans RED_G

❖ par **récurrence** sur (u, v) ordonné par $\rightarrow^* \times_{\text{lexicogr.}} \rightarrow^*$
(les deux sont dans SN, par (CR1))... comme d'habitude

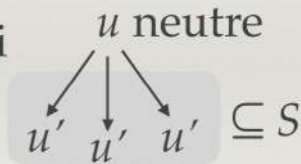
(C) $(\mathbf{C}u)v \rightarrow \mathbf{C}(\lambda k . u(\lambda f . k(fv)))$

(η C) $\mathbf{C}(\lambda k . ku) \rightarrow u$

Un autre lemme utile

(CR1) $S \subseteq \text{SN}$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


❖ ... nous devons donc démontrer que
 $(Cu)v \in \text{RED}_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$

(η C) $C(\lambda k . ku) \rightarrow u$

(H) $u \in \text{RED}_{\neg\neg(F \Rightarrow G)}$

(H') $v \in \text{RED}_F$

(Hrec) pour tout terme $w \in \text{RED}_{\neg\neg G}$, $Cw \in \text{RED}_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in \text{RED}_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in \text{RED}_{F \Rightarrow G}$

Un autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{array}{c} u \\ \swarrow \downarrow \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$

❖ ... nous devons donc démontrer que
 $(Cu)v \in RED_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 $(\eta C) C(\lambda k . ku) \rightarrow u$

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

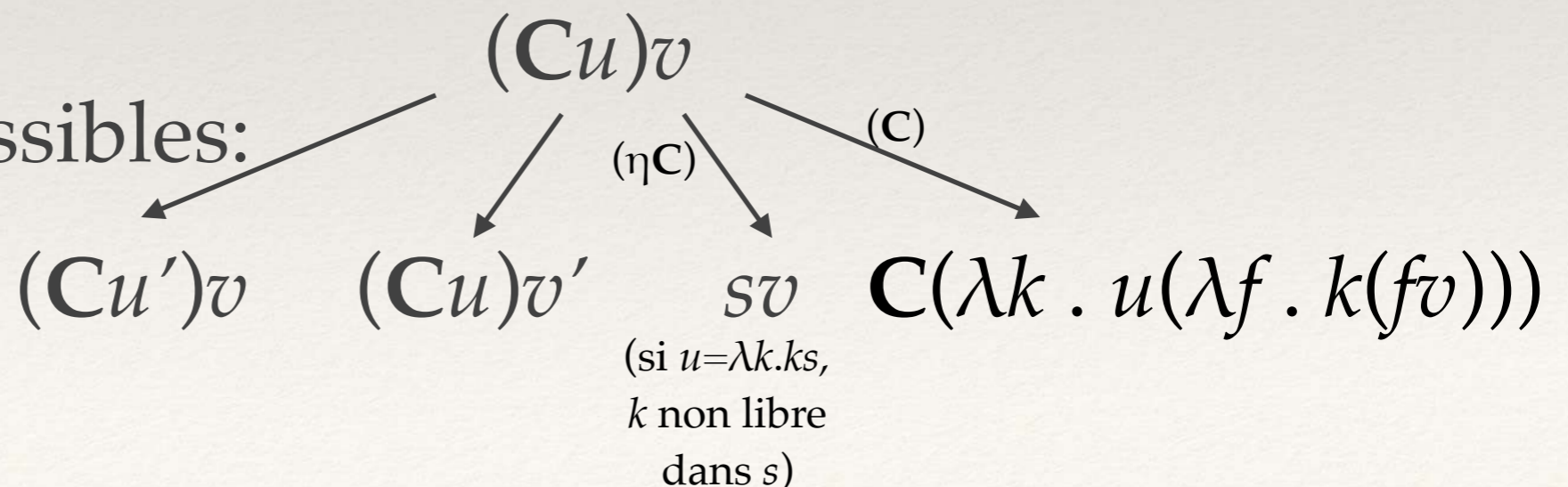
(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

❖ Il y a quatre cas

de réduction possibles:

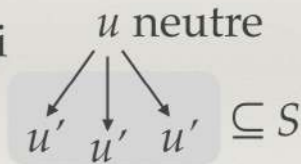


Un autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$



$\subseteq S$

❖ ... nous devons donc démontrer que $(Cu)v \in RED_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 (ηC) $C(\lambda k . ku) \rightarrow u$

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

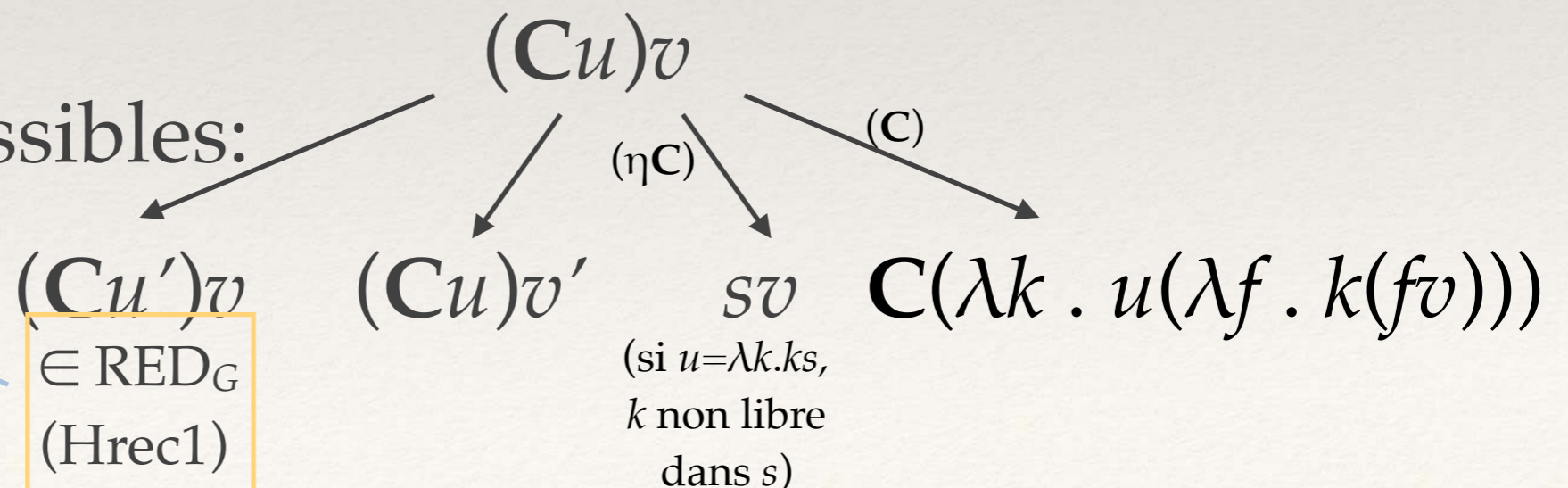
(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduct u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduct v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

❖ Il y a quatre cas de réduction possibles:



Un autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{array}{c} u \\ \swarrow \downarrow \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$

❖ ... nous devons donc démontrer que
 $(Cu)v \in RED_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 $(\eta C) C(\lambda k . ku) \rightarrow u$

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

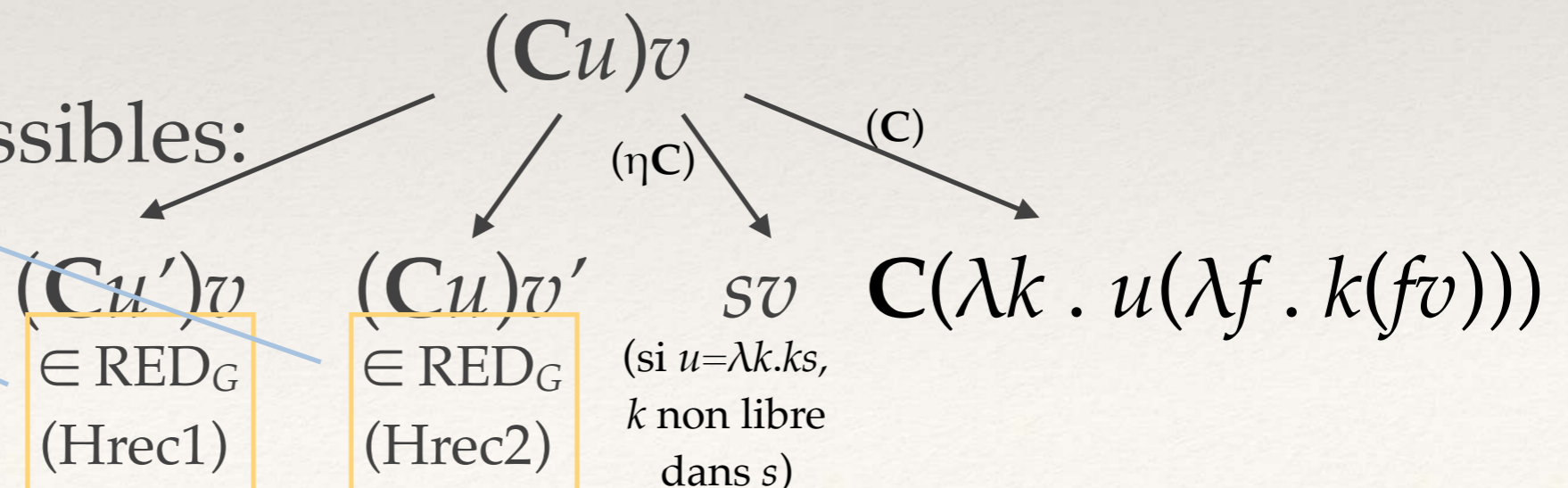
(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

❖ Il y a quatre cas
de réduction possibles:



Un autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{matrix} & u & \\ & \swarrow \downarrow \searrow & \\ u' & & u' \end{matrix} \subseteq S$

❖ ... nous devons donc démontrer que
 $(Cu)v \in RED_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 $(\eta C) C(\lambda k . ku) \rightarrow u$

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

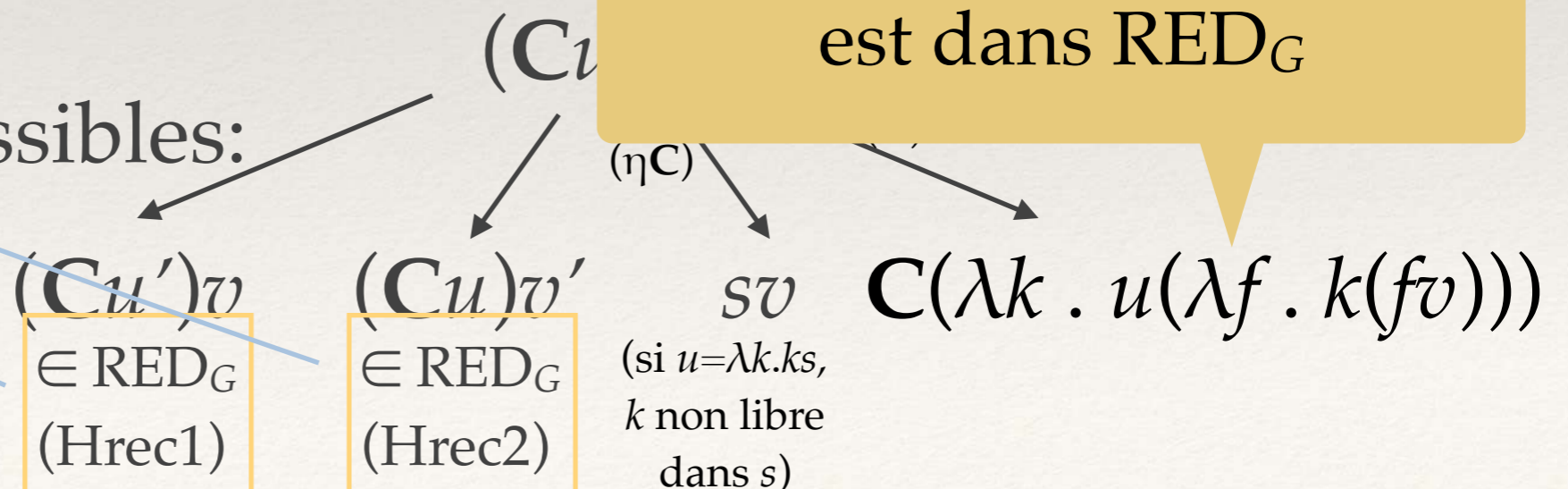
(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

❖ Il y a quatre cas de réduction possibles:



(C'est assez impressionnant à démontrer...)
 $C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

?

$C(\lambda k . u(\lambda f . k(fv))) \in RED_G$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

?

 $\lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G}$

(Hrec)

 $C(\lambda k . u(\lambda f . k(fv))) \in RED_G$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

?

$\forall s \in RED_{\neg G}, u(\lambda f . s(fv)) \in RED_{\perp}$

Lemme C

$\lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G}$

(Hrec)

$C(\lambda k . u(\lambda f . k(fv))) \in RED_G$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
alors $\lambda x.s \in RED_{F \Rightarrow G}$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

$$\begin{array}{c}
 \frac{?}{u \in RED_{\neg\neg(F \Rightarrow G)}} \quad \frac{?}{\forall s \in RED_{\neg G}, \lambda f . s(fv) \in RED_{\neg(F \Rightarrow G)}} \quad \text{(déf. de } RED_{\neg\neg(F \Rightarrow G)}\text{)} \\
 \hline
 \forall s \in RED_{\neg G}, u(\lambda f . s(fv)) \in RED_{\perp} \\
 \hline
 \lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G} \quad \text{Lemme C} \\
 \hline
 C(\lambda k . u(\lambda f . k(fv))) \in RED_G \quad \text{(Hrec)}
 \end{array}$$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
alors $\lambda x.s \in RED_{F \Rightarrow G}$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

?

$u \in RED_{\neg\neg(F \Rightarrow G)}$ (H)

$\forall s \in RED_{\neg G}, \lambda f . s(fv) \in RED_{\neg(F \Rightarrow G)}$

(déf. de $RED_{\neg\neg(F \Rightarrow G)}$)

$\forall s \in RED_{\neg G}, u(\lambda f . s(fv)) \in RED_{\perp}$

Lemme C

$\lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G}$

(Hrec)

$C(\lambda k . u(\lambda f . k(fv))) \in RED_G$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
alors $\lambda x.s \in RED_{F \Rightarrow G}$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

?

$$\begin{array}{c}
 \frac{u \in RED_{\neg\neg(F \Rightarrow G)} \quad \frac{\frac{\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, s(tv) \in RED_{\perp}}{\forall s \in RED_{\neg G}, \lambda f . s(fv) \in RED_{\neg(F \Rightarrow G)}} \text{Lemme C}}{\forall s \in RED_{\neg G}, u(\lambda f . s(fv)) \in RED_{\perp}} \text{(déf. de } RED_{\neg\neg(F \Rightarrow G)}\text{)}}}{\lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G}} \text{Lemme C}} \\
 \frac{\lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G}}{C(\lambda k . u(\lambda f . k(fv))) \in RED_G} \text{(Hrec)}
 \end{array}$$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
alors $\lambda x.s \in RED_{F \Rightarrow G}$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

$\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G},$

$tv \in RED_G$ (déf. de $RED_{F \Rightarrow G} + (H')$)

donc $s(tv) \in RED_{\perp}$ (déf. de $RED_{\neg G}$)

(déf. de $RED_{F \Rightarrow G}$, de $RED_{\neg G} + (H')$)

$\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, s(tv) \in RED_{\perp}$

Lemme C

$u \in RED_{\neg\neg(F \Rightarrow G)}$ (H)

$\forall s \in RED_{\neg G}, \lambda f . s(fv) \in RED_{\neg(F \Rightarrow G)}$

(déf. de $RED_{\neg\neg(F \Rightarrow G)}$)

$\forall s \in RED_{\neg G}, u(\lambda f . s(fv)) \in RED_{\perp}$

Lemme C

$\lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G}$

(Hrec)

$C(\lambda k . u(\lambda f . k(fv))) \in RED_G$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
alors $\lambda x.s \in RED_{F \Rightarrow G}$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

Vous avez sans doute vu le parallèle avec le **typage**:

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

$\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G},$

$tv \in RED_G$ (déf. de $RED_{F \Rightarrow G} + (H')$)

donc $s(tv) \in RED_{\perp}$ (déf. de $RED_{\neg G}$)

(déf. de $RED_{F \Rightarrow G}$, de $RED_{\neg G} + (H')$)

$\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, s(tv) \in RED_{\perp}$

Lemme C

$u \in RED_{\neg\neg(F \Rightarrow G)}$ (H)

$\forall s \in RED_{\neg G}, \lambda f . s(fv) \in RED_{\neg(F \Rightarrow G)}$

(déf. de $RED_{\neg\neg(F \Rightarrow G)}$)

$\forall s \in RED_{\neg G}, u(\lambda f . s(fv)) \in RED_{\perp}$

Lemme C

$\lambda k . u(\lambda f . k(fv)) \in RED_{\neg\neg G}$

(Hrec)

$C(\lambda k . u(\lambda f . k(fv))) \in RED_G$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
alors $\lambda x.s \in RED_{F \Rightarrow G}$

(C'est assez impressionnant à démontrer...)

$C(\lambda k . u(\lambda f . k(fv)))$ est dans RED_G

Vous avez sans doute vu le parallèle avec le **typage**:

... sous les hypothèses:

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

$$\begin{array}{c}
 \Gamma \vdash v : F \\
 \downarrow \text{(affaiblissement)} \\
 \Gamma \vdash v : F \\
 \hline
 \Gamma \vdash u : \neg\neg(F \Rightarrow G) \\
 \downarrow \text{(affaiblissement)} \\
 \Gamma, k : \neg\neg(F \Rightarrow G) \vdash u : \neg\neg(F \Rightarrow G) \\
 \hline
 \Gamma, k : \neg\neg(F \Rightarrow G) \vdash \lambda f . k(fv) : \neg(F \Rightarrow G) \\
 \hline
 \Gamma \vdash \lambda k . u(\lambda f . k(fv)) : \neg\neg G \\
 \hline
 \Gamma \vdash C(\lambda k . u(\lambda f . k(fv))) : G
 \end{array}$$

$\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G} \dots$
 $\dots \vdash k : \neg G$
 $\dots \vdash fv : G$
 $\dots \vdash \lambda f . k(fv) : \neg(F \Rightarrow G)$
 $\dots \vdash u : \neg\neg(F \Rightarrow G)$
 $\dots \vdash \lambda k . u(\lambda f . k(fv)) : \neg\neg G$
 $\dots \vdash C(\lambda k . u(\lambda f . k(fv))) : G$

(Ax) $\Gamma \vdash v : F$
 (Ax) $\dots \vdash k : \neg G$
 (Ax) $\dots \vdash fv : G$
 (Ax) $\dots \vdash \lambda f . k(fv) : \neg(F \Rightarrow G)$
 (Ax) $\dots \vdash u : \neg\neg(F \Rightarrow G)$
 (Ax) $\dots \vdash \lambda k . u(\lambda f . k(fv)) : \neg\neg G$
 (Ax) $\dots \vdash C(\lambda k . u(\lambda f . k(fv))) : G$

(=>E) $\Gamma, k : \neg\neg(F \Rightarrow G), f : F \Rightarrow G \vdash k(fv) : \perp$
 (=>E) $\Gamma, k : \neg\neg(F \Rightarrow G) \vdash \lambda f . k(fv) : \neg(F \Rightarrow G)$
 (=>E) $\Gamma \vdash \lambda k . u(\lambda f . k(fv)) : \neg\neg G$
 (=>E) $\Gamma \vdash C(\lambda k . u(\lambda f . k(fv))) : G$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$ alors $\lambda x.s \in RED_{F \Rightarrow G}$

Retour à la preuve...

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{array}{c} u \\ \swarrow \downarrow \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$

❖ ... nous devons donc démontrer que $(Cu)v \in RED_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 $(\eta C) C(\lambda k . ku) \rightarrow u$

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

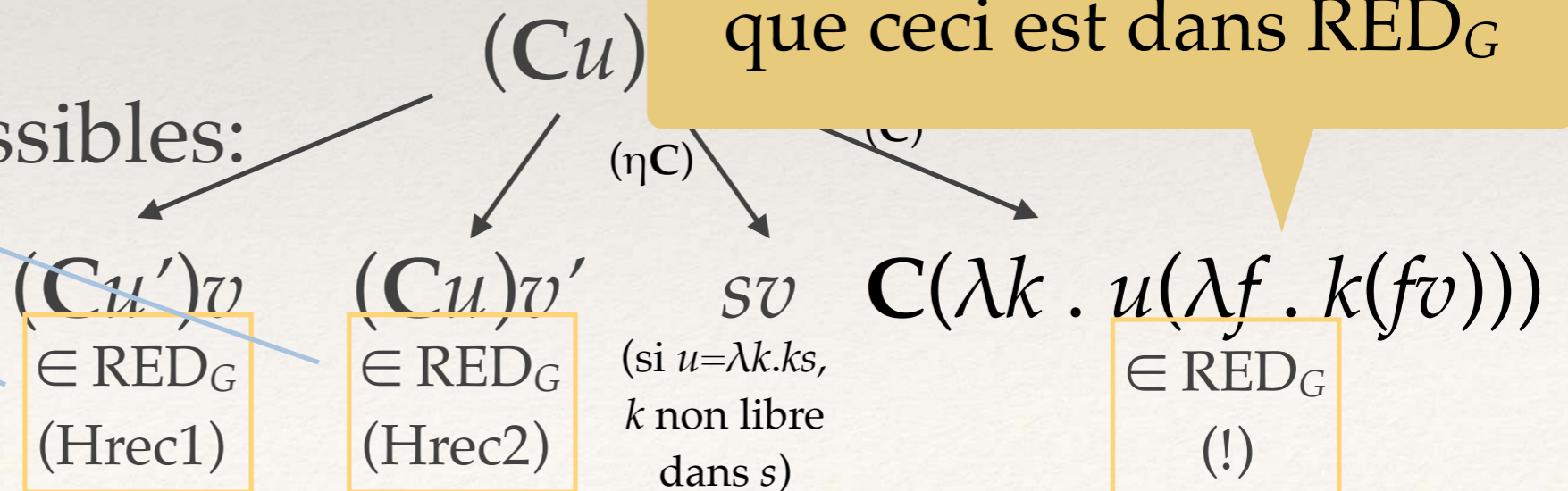
(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout réduit v' de v

❖ Il y a quatre cas de réduction possibles:

nous venons de montrer que ceci est dans RED_G



Retour à la preuve...

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{matrix} & u & \\ & \swarrow \downarrow \searrow & \\ u' & & u' \end{matrix} \subseteq S$

❖ ... nous devons donc démontrer que
 $(Cu)v \in RED_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 $(\eta C) C(\lambda k . ku) \rightarrow u$

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout $u' \in RED_{\neg\neg(F \Rightarrow G)}$, $(Cu')v \in RED_{F \Rightarrow G}$

montrons maintenant
 que ceci est dans RED_G

❖ Il y a quatre cas
 de réduction possibles.

$(Cu')v$
 $\in RED_G$
 (Hrec1)

$(Cu)v'$
 $\in RED_G$
 (Hrec2)

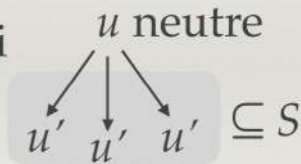
sv
 (si $u = \lambda k . ks$,
 k non libre
 dans s)

$C(\lambda k . u(\lambda f . k(fv)))$
 $\in RED_G$
 (!)

Le 4ème cas

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


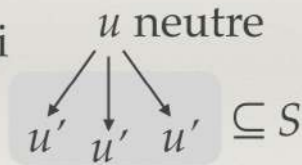
- ❖ ... montrons que si $u = \lambda k . ks \in \text{RED}_{\neg\neg H}$
où k n'est pas libre dans s ,
alors $s \in \text{RED}_H$

(ici $H = F \Rightarrow G$)

Le 4ème cas

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


- ❖ ... montrons que si $u = \lambda k.k s \in RED_{\neg\neg H}$
où k n'est pas libre dans s ,
alors $s \in RED_H$

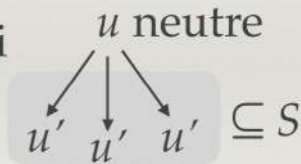
(ici $H = F \Rightarrow G$)

- ❖ Ceci est par récurrence sur H

Le 4ème cas

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

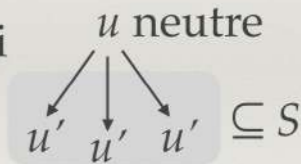
(CR3) si u neutre alors $u \in S$


- ❖ ... montrons que si $u = \lambda k.k s \in \text{RED}_{\neg\neg H}$ (ici $H = F \Rightarrow G$)
où k n'est pas libre dans s ,
alors $s \in \text{RED}_H$
- ❖ Ceci est par récurrence sur H
- ❖ Dans le cas d'un type flèche, on utilise le Lemme C et la définition de RED, comme au 3ème cas
(voir poly types.pdf, théorème 3, claim (**), p.17).

Le 4ème cas

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si  alors $u \in S$

- ❖ ... montrons que si $u = \lambda k.k s \in \text{RED}_{\neg\neg H}$ (ici $H = F \Rightarrow G$)
où k n'est pas libre dans s ,
alors $s \in \text{RED}_H$
- ❖ Ceci est par récurrence sur H
- ❖ Dans le cas d'un type flèche, on utilise le Lemme C et la définition de RED, comme au 3ème cas
(voir poly types.pdf, théorème 3, claim (**), p.17).
- ❖ Je vous l'épargne!

Retour à la preuve...

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{array}{c} u \\ \swarrow \downarrow \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$

❖ ... nous devons donc démontrer que $(Cu)v \in RED_G$, sachant que

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 $(\eta C) C(\lambda k . ku) \rightarrow u$

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

(Hrec2) pour tout $u' \in RED_{\neg\neg(F \Rightarrow G)}$, $(Cu')v \in RED_{F \Rightarrow G}$

montrons maintenant que ceci est dans RED_G

❖ Il y a quatre cas de réduction possibles.

$(Cu')v \in RED_G$
(Hrec1)

$(Cu)v' \in RED_G$
(Hrec2)

sv
(si $u = \lambda k . ks$,
 k non libre dans s)

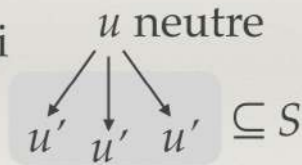
$C(\lambda k . u(\lambda f . k(fv))) \in RED_G$
(!)

Retour à la preuve...

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$



❖ ... nous devons donc démontrer que

$(Cu)v \in RED_G$, sachant que

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

le v , $(Cu)v' \in RED_{F \Rightarrow G}$

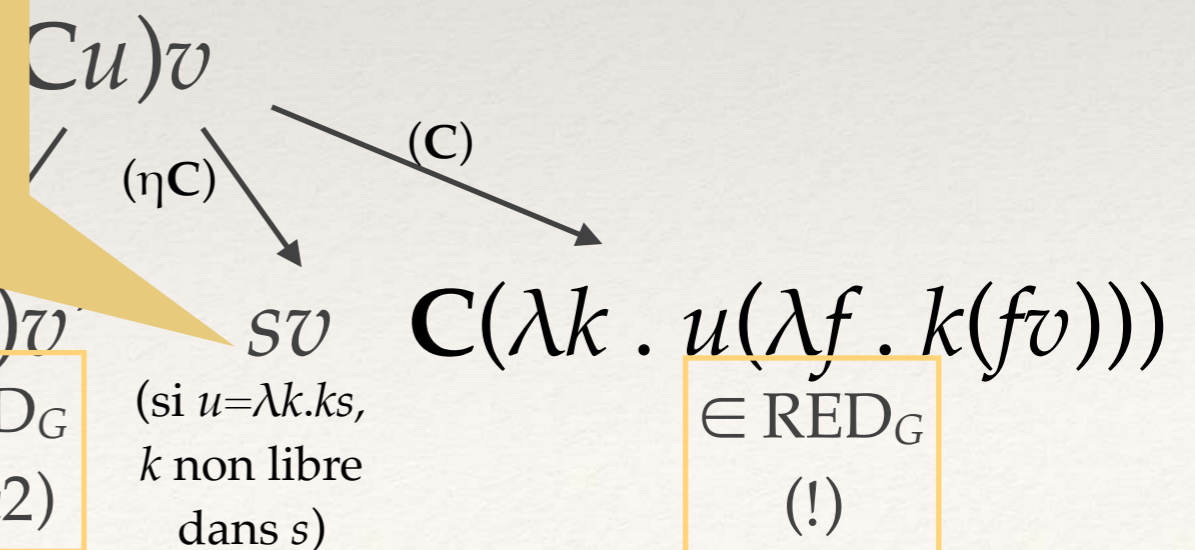
(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$

(ηC) $C(\lambda k . ku) \rightarrow u$

par le transparent précédent,

$s \in RED_{F \Rightarrow G}$,

et $v \in RED_F$ par (H'), donc...



$(Cu')v$
 $\in RED_G$
 (Hrec1)

$(Cu)v$
 $\in RED_G$
 (Hrec2)

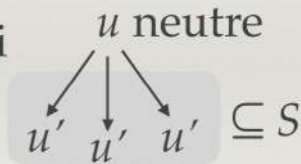
$C(\lambda k . u(\lambda f . k(fv)))$
 $\in RED_G$
 (!)

Retour à la preuve...

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$



$\subseteq S$

❖ ... nous devons donc démontrer que

$(Cu)v \in RED_G$, sachant que

(H) $u \in RED_{\neg\neg(F \Rightarrow G)}$

(H') $v \in RED_F$

(Hrec) pour tout terme $w \in RED_{\neg\neg G}$, $Cw \in RED_G$

(Hrec1) pour tout réduit u' de u , $(Cu')v \in RED_{F \Rightarrow G}$

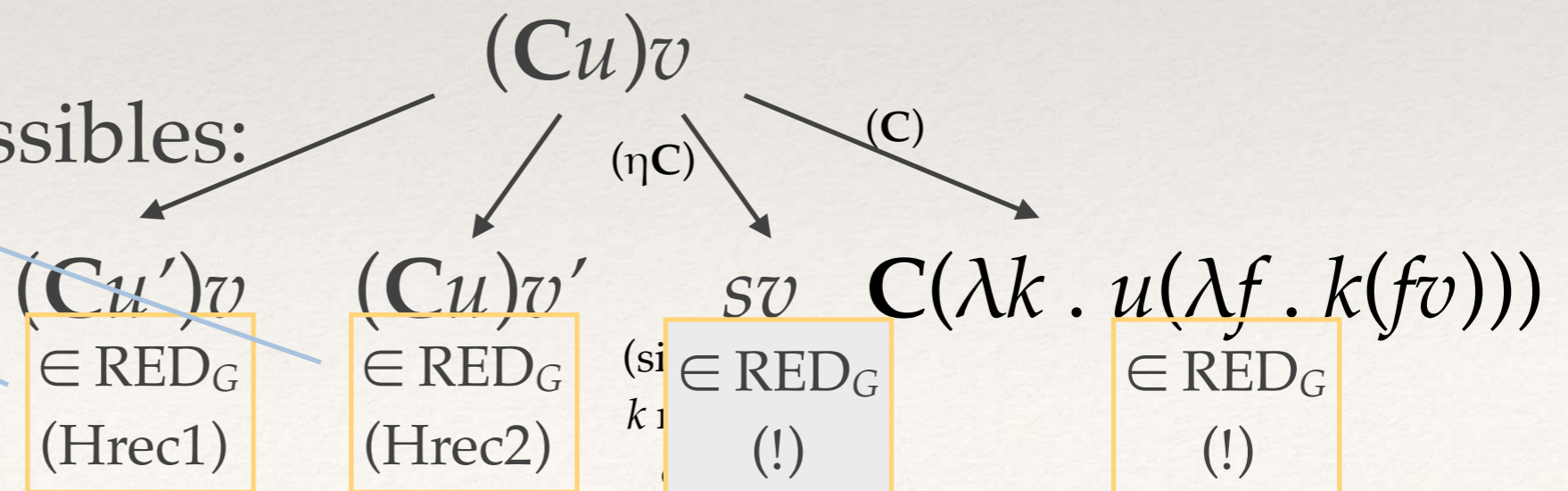
(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$

(η C) $C(\lambda k . ku) \rightarrow u$

❖ Il y a quatre cas

de réduction possibles:

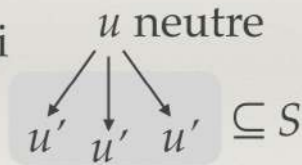


Retour à la preuve...

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$



❖ ... nous devons donc démontrer que

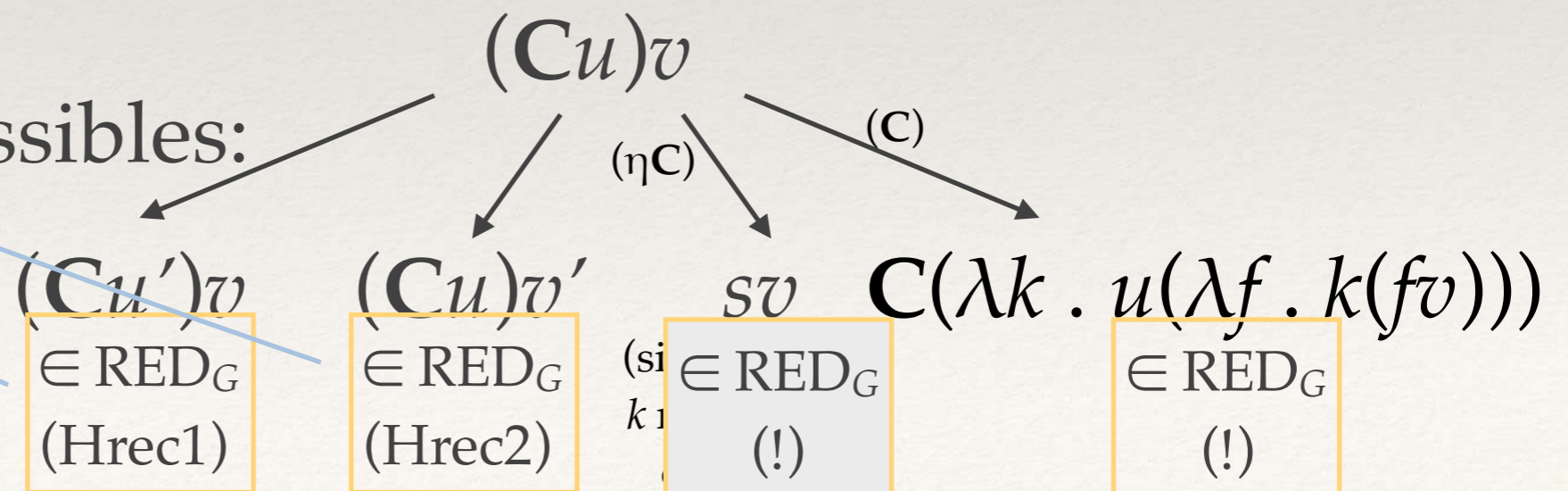
(C) $(Cu)v \rightarrow C(\lambda k . u(\lambda f . k(fv)))$
 (ηC) $C(\lambda k . ku) \rightarrow u$

par (CR3) au type G ,
 comme $(Cu)v$ est neutre,
 il est dans RED_G .

Ceci termine le cas $F \Rightarrow G$ de...

(Hrec2) pour tout réduit v' de v , $(Cu)v' \in RED_{F \Rightarrow G}$

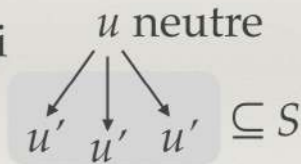
❖ Il y a quatre cas
 de réduction possibles:



... l'autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$


- ❖ **Lemme D.** Pour tout type F , si $u \in RED_{\neg\neg F}$,
alors $Cu \in RED_F$

... l'autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{array}{c} u \\ \swarrow \downarrow \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$

❖ **Lemme D.** Pour tout type F , si $u \in RED_{\neg\neg F}$,
 alors $Cu \in RED_F$

❖ Résumé: par récurrence sur F ;
 pour les types flèches, on utilise (CR3):
 4 cas possibles, et un raisonnement infernal...
 qui ressemblait à une dérivation de typage

$$\begin{array}{c}
 \frac{\frac{\frac{u \in RED_{\neg\neg(F \Rightarrow G)}}{(H)} \quad \frac{\frac{\frac{\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, tv \in RED_G \text{ (déf. de } RED_{F \Rightarrow G} + (H'))}{\text{donc } s(tv) \in RED_{\perp} \text{ (déf. de } RED_{\neg G})}}{\forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, s(tv) \in RED_{\perp}} \text{ (déf. de } RED_{F \Rightarrow G}, \text{ de } RED_{\neg G} + (H'))}{\forall s \in RED_{\neg G}, \lambda f. s(fv) \in RED_{\neg(F \Rightarrow G)}} \text{ Lemme C}}{\forall s \in RED_{\neg G}, u(\lambda f. s(fv)) \in RED_{\perp}} \text{ (déf. de } RED_{\neg\neg(F \Rightarrow G)})}{\lambda k. u(\lambda f. k(fv)) \in RED_{\neg\neg G}} \text{ Lemme C}} \\
 \frac{\lambda k. u(\lambda f. k(fv)) \in RED_{\neg\neg G}}{C(\lambda k. u(\lambda f. k(fv))) \in RED_G} \text{ (Hrec)}
 \end{array}$$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
 alors $\lambda x.s \in RED_{F \Rightarrow G}$

... l'autre lemme utile

(CR1) $S \subseteq SN$

(CR2) si $u \in S \rightarrow u'$ alors $u' \in S$

(CR3) si u neutre alors $u \in S$
 $\begin{array}{c} u \\ \swarrow \downarrow \searrow \\ u' \quad u' \quad u' \end{array} \subseteq S$

❖ **Lemme D.** Pour tout type F , si $u \in RED_{\neg\neg F}$,
 alors $Cu \in RED_F$

❖ Résumé: par récurrence sur F ;
 pour les types flèches, on utilise (CR3):
 4 cas possibles, et un raisonnement infernal...
 qui ressemblait à une dérivation de typage

$$\begin{array}{c}
 \forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, \\
 tv \in RED_G \text{ (déf. de } RED_{F \Rightarrow G} + (H')) \\
 \text{donc } s(tv) \in RED_{\perp} \text{ (déf. de } RED_{\neg G}) \\
 \hline
 \forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, s(tv) \in RED_{\perp} \text{ (déf. de } RED_{F \Rightarrow G}, \text{ de } RED_{\neg G} + (H')) \\
 \hline
 \forall s \in RED_{\neg G}, \forall t \in RED_{F \Rightarrow G}, s(ftv) \in RED_{\perp} \text{ (Lemme C)} \\
 \hline
 \forall s \in RED_{\neg G}, \lambda f. s(ftv) \in RED_{\neg(F \Rightarrow G)} \text{ (déf. de } RED_{\neg(F \Rightarrow G)}) \\
 \hline
 \forall s \in RED_{\neg G}, u(\lambda f. s(ftv)) \in RED_{\perp} \text{ (Lemme C)} \\
 \hline
 \lambda k. u(\lambda f. k(ftv)) \in RED_{\neg\neg G} \text{ (Hrec)} \\
 \hline
 C(\lambda k. u(\lambda f. k(ftv))) \in RED_G
 \end{array}$$

Lemme C. Si $(\forall v \in RED_F, s[x:=v] \in RED_G)$
 alors $\lambda x.s \in RED_{F \Rightarrow G}$

Le raisonnement infernal

$$\begin{array}{c}
 \Gamma \vdash v : F \\
 \downarrow \text{(affaiblissement)} \\
 \dots \vdash v : F \\
 \hline
 \dots \vdash f : F \Rightarrow G \text{ (Ax)} \quad \dots \vdash v : F \text{ (}\Rightarrow\text{E)} \\
 \hline
 \dots \vdash k : \neg G \text{ (Ax)} \quad \dots \vdash fv : G \text{ (}\Rightarrow\text{E)} \\
 \hline
 \Gamma \vdash u : \neg\neg(F \Rightarrow G) \quad \Gamma, k : \neg G, f : F \Rightarrow G \vdash k(fv) : \perp \text{ (}\Rightarrow\text{I)} \\
 \downarrow \text{(affaiblissement)} \quad \hline
 \Gamma, k : \neg G \vdash u : \neg\neg(F \Rightarrow G) \quad \Gamma, k : \neg G \vdash \lambda f. k(fv) : \neg(F \Rightarrow G) \text{ (}\Rightarrow\text{E)} \\
 \hline
 \Gamma, k : \neg G \vdash u(\lambda f. k(fv)) : \perp \text{ (}\Rightarrow\text{E)} \\
 \hline
 \Gamma \vdash \lambda k. u(\lambda f. k(fv)) : \neg\neg G \text{ (}\Rightarrow\text{I)} \\
 \hline
 \Gamma \vdash C(\lambda k. u(\lambda f. k(fv))) : G \text{ (}\neg\neg\text{E)}
 \end{array}$$

La dérivation de typage analogue

Normalisation forte

- ❖ **Def:** $\theta \in \underline{\text{RED}}_\Gamma$ ssi pour tout $x:F$ dans Γ , $\theta(x) \in \text{RED}_F$
- ❖ **Thm.** si $\Gamma \vdash u : G$ dérivable alors
pour toute $\theta \in \underline{\text{RED}}_\Gamma$, $u\theta \in \text{RED}_G$.

Normalisation forte

- ❖ **Def:** $\theta \in \underline{RED}_\Gamma$ ssi pour tout $x:F$ dans Γ , $\theta(x) \in RED_F$
- ❖ **Thm.** si $\Gamma \vdash u : G$ dérivable alors
pour toute $\theta \in \underline{RED}_\Gamma$, $u\theta \in RED_G$.
- ❖ Preuve: par récurrence sur la dérivation de $\Gamma \vdash u : G$,
comme la dernière fois;
on utilise la déf. de $RED_{F \Rightarrow G}$ dans le cas de (\Rightarrow E), le
Lemme C pour (\Rightarrow I),
et le Lemme D pour ($\neg\neg$ E).

$$\frac{\Gamma \vdash u : \neg\neg G}{\Gamma \vdash Cu : G} (\neg\neg E)$$

Lemme D. Pour tout type F , si $u \in RED_{\neg\neg F}$,
alors $Cu \in RED_F$

Normalisation forte

- ❖ **Def:** $\theta \in \underline{\text{RED}}_\Gamma$ ssi pour tout $x:F$ dans Γ , $\theta(x) \in \text{RED}_F$
- ❖ **Thm.** si $\Gamma \vdash u : G$ dérivable alors
pour toute $\theta \in \underline{\text{RED}}_\Gamma$, $u\theta \in \text{RED}_G$.
- ❖ **Corollaire.** Tout $\lambda\mathbf{C}$ -terme simplement typable est fortement normalisable.

Pourquoi « classique »?

- ❖ On peut démontrer que le système de déduction naturelle sous-jacent à λC est **correct** et **complet** pour l'interprétation dans les booléens de la dernière fois

$$\frac{\Gamma \vdash \neg\neg G}{\Gamma \vdash G} (\neg\neg E)$$

classique

« logique minimale en déduction naturelle »

n'importe quoi implique vrai

\Rightarrow	0	1
0	1	1
1	0	1

faux implique n'importe quoi

- ❖ On définit une **sémantique...** à 2 valeurs de vérité (0=faux, 1=vrai)
 $\llbracket F \Rightarrow G \rrbracket \rho = \llbracket F \rrbracket \rho \Rightarrow \llbracket G \rrbracket \rho$, voir: ➔
(c'est quand même plus simple que les candidats de réductibilité!)
- ❖ $\llbracket A \rrbracket \rho = \rho(A)$ — ρ est un **environnement** qui à chaque formule atomique associe sa valeur de vérité

- ❖ Ca prend du temps...
(omis)

Voilà

- ❖ ... pour les logiques propositionnelles
- ❖ Après les vacances, nous passerons à:
 - la logique et l'arithmétique du 1er ordre
 - l'arithmétique de 2nd ordre et le système F
 - enfin à certaines considérations théoriques sur l'implémentation de machines de réduction pour le λ -calcul
- ❖ Bonnes vacances, et revenez sains et saufs!