Une formalisation en Rocq/Coq des réseaux de preuve de la logique linéaire

Rémi Di Guardia

IRIF (Université Paris Cité)

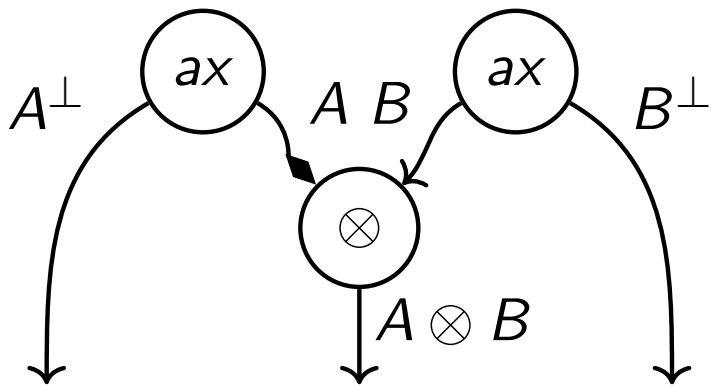
Strasbourg, 3 Avril 2025

Introduction

Contexte

- Théorie de la démonstration - Calcul des séquents
- Logique linéaire
- Réseaux de preuve = autre représentation des preuves que les arbres du calcul des séquents

$$\dfrac{\dfrac{}{\vdash A^{\perp}, A}\,(ax) \qquad \dfrac{}{\vdash B^{\perp}, B}\,(ax)}{\vdash A^{\perp}, A \otimes B^{\perp}, B}\,(\otimes)$$



Pourquoi ces réseaux ne sont pas déjà formalisés ?

Déjà des formalisations de la logique linéaire . . .

- Rocq [Lau17; PW99; Xav+18; CL; PC; Sad03]
- Abella [CLR19; CLR17]
- Isabelle [KP95; Gro95]

. . . mais **jamais** avec les réseaux de preuve !

Pourquoi ces réseaux ne sont pas déjà formalisés ?

Déjà des formalisations de la logique linéaire . . .

- Rocq [Lau17; PW99; Xav+18; CL; PC; Sad03]
- Abella [CLR19; CLR17]
- Isabelle [KP95; Gro95]

. . . mais **jamais** avec les réseaux de preuve !

Quelles difficultés à résoudre ?

- **Manipulation explicites de multigraphes** et de leurs isomorphismes

Pourquoi ces réseaux ne sont pas déjà formalisés ?

Déjà des formalisations de la logique linéaire . . .

- Rocq [Lau17; PW99; Xav+18; CL; PC; Sad03]
- Abella [CLR19; CLR17]
- Isabelle [KP95; Gro95]

. . . mais **jamais** avec les réseaux de preuve !

Quelles difficultés à résoudre ?

- **Manipulation explicites de multigraphes** et de leurs isomorphismes
- Définition **complexe**, déjà par strates sur papier (structures et réseaux)

Pourquoi ces réseaux ne sont pas déjà formalisés ?

Déjà des formalisations de la logique linéaire . . .

- Rocq [Lau17; PW99; Xav+18; CL; PC; Sad03]
- Abella [CLR19; CLR17]
- Isabelle [KP95; Gro95]

. . . mais **jamais** avec les réseaux de preuve !

Quelles difficultés à résoudre ?

- **Manipulation explicites de multigraphes** et de leurs isomorphismes
- Définition **complexe**, déjà par strates sur papier (structures et réseaux)
- Plusieurs définitions équivalentes sur papier avec des **variations** : laquelle est la plus adaptée à être formalisée, et sous quelle forme ?

Pourquoi ces réseaux ne sont pas déjà formalisés ?

Déjà des formalisations de la logique linéaire . . .

- Rocq [Lau17; PW99; Xav+18; CL; PC; Sad03]
- Abella [CLR19; CLR17]
- Isabelle [KP95; Gro95]

. . . mais **jamais** avec les réseaux de preuve !

Quelles difficultés à résoudre ?

- **Manipulation explicites de multigraphes** et de leurs isomorphismes
- Définition **complexe**, déjà par strates sur papier (structures et réseaux)
- Plusieurs définitions équivalentes sur papier avec des **variations** : laquelle est la plus adaptée à être formalisée, et sous quelle forme ?
- Arguments **géométriques** / **graphiques** sur papier, avec des dessins

Pourquoi formaliser ces réseaux maintenant ?

- Formaliser ces résultats est justement intéressant à cause des libertés prises sur papier !
- Librairie récente **GraphTheory** en Rocq (1ère version en 2020) par Damien Pous et Christian Doczkal, bâtie sur *Mathematical Components* et *SSReflect*
  - manipulation de graphes, avec des résultats plutôt difficiles (sur la treewidth, les mineurs, . . .)
  - contient les multigraphes avec les opérations dont on a besoin : ajout et retrait de sommets, sous-graphes, isomorphismes, associativité de l'union disjointe à isomorphisme près, . . .

Calcul des séquents

Formules et **Connecteurs** : $A, B, A \wedge B, A \vee B, \neg A, \ldots$

Séquents : Hypothèse(s) et conclusion(s), « *énoncés* »

$$A, B \vdash A \wedge B$$

Calcul des séquents

Formules et **Connecteurs** : $A, B, A \wedge B, A \vee B, \neg A, \ldots$

Séquents : Hypothèse(s) et conclusion(s), « *énoncés* »

$$A, B \vdash A \wedge B$$

Règles : Prouve un séquent à partir d'autres séquents

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \wedge B}\ (\wedge) \qquad \frac{\Gamma \vdash A, B}{\Gamma \vdash A \vee B}\ (\vee) \qquad \frac{}{A \vdash A}\ (ax)$$

Calcul des séquents

Formules et **Connecteurs** : $A, B, A \wedge B, A \vee B, \neg A, \ldots$

Séquents : Hypothèse(s) et conclusion(s), « *énoncés* »

$$A, B \vdash A \wedge B$$

Règles : Prouve un séquent à partir d'autres séquents

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \wedge B}\ (\wedge) \qquad \frac{\Gamma \vdash A, B}{\Gamma \vdash A \vee B}\ (\vee) \qquad \frac{}{A \vdash A}\ (ax)$$

Arbres de preuve :

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\ (ax) \qquad \dfrac{}{B \vdash B}\ (ax)}{A, B \vdash A \wedge B}\ (\wedge) \qquad \dfrac{}{C \vee D \vdash C \vee D}\ (ax)}{A, B, C \vee D \vdash (A \wedge B) \wedge (C \vee D)}\ (\wedge)$$

Logique Classique

Formules :

$$A, B := X \mid A \wedge B \mid A \vee B \mid \dots$$

Règles :

$$\frac{\Gamma \vdash A, \Sigma \qquad \Delta \vdash B, \Theta}{\Gamma, \Delta \vdash A \wedge B, \Sigma, \Theta}\ (\wedge) \qquad \frac{\Gamma \vdash A, B, \Sigma}{\Gamma \vdash A \vee B, \Sigma}\ (\vee)$$

$$\frac{\Gamma \vdash A, \Sigma \qquad \Gamma \vdash B, \Sigma}{\Gamma \vdash A \wedge B, \Sigma}\ (\wedge) \qquad \frac{\Gamma \vdash A, \Sigma}{\Gamma \vdash A \vee B, \Sigma}\ (\vee)$$

(et d'autres règles)

Logique Linéaire

Formules :

$$A, B := X \mid A \otimes B \mid A \,\&\, B \mid A \parr B \mid A \oplus B \mid \dots$$

Règles :

$$\frac{\Gamma \vdash A, \Sigma \qquad \Delta \vdash B, \Theta}{\Gamma, \Delta \vdash A \otimes B, \Sigma, \Theta}\ (\otimes) \qquad \frac{\Gamma \vdash A, B, \Sigma}{\Gamma \vdash A \parr B, \Sigma}\ (\parr)$$

$$\frac{\Gamma \vdash A, \Sigma \qquad \Gamma \vdash B, \Sigma}{\Gamma \vdash A \,\&\, B, \Sigma}\ (\&) \qquad \frac{\Gamma \vdash A, \Sigma}{\Gamma \vdash A \oplus B, \Sigma}\ (\oplus)$$

(et d'autres règles)

- Généralise la logique classique et la logique intuitionniste
- Chaque formule est utilisée exactement une fois dans une preuve, comme une ressource : pas possible d'avoir $A \otimes A$ sachant A

Plan

- **Logique Linéaire Multiplicative**
  - Calcul des séquents
  - Réseaux de preuve
- **Formalisation des réseaux de preuve**
  - Structures de preuve
  - Temps de calcul

Logique Linéaire Multiplicative

Formules :

$$A, B := X \mid \overset{\text{non}}{X^{\perp}} \mid A \overset{\text{et}}{\otimes} B \mid A \overset{\text{ou}}{\parr} B$$

$X \; ; \; X^{\perp} \parr Y \; ; \; (X^{\perp} \parr Y^{\perp}) \parr Z$

Logique Linéaire Multiplicative

Formules :

$$A, B := X \mid X^{\perp}{}^{\text{non}} \mid A \overset{\text{et}}{\otimes} B \mid A \overset{\text{ou}}{\parr} B$$

$$X \; ; \; X^{\perp} \parr Y \; ; \; (X^{\perp} \parr Y^{\perp}) \parr Z$$

Séquents :

$$\vdash A_1 \overset{\text{ou}}{,} A_2, \ldots, A_n$$

$$\vdash X \otimes Y, (X^{\perp} \parr Y^{\perp}) \parr Z, Z^{\perp}$$

Logique Linéaire Multiplicative

Formules :

$$A, B := X \mid X^{\perp} \mid A \overset{\text{et}}{\otimes} B \mid A \overset{\text{ou}}{\parr} B$$

$$X \; ; \; X^{\perp} \parr Y \; ; \; (X^{\perp} \parr Y^{\perp}) \parr Z$$

Séquents :

$$\vdash A_1 \overset{\text{ou}}{,} A_2, \ldots, A_n$$

$$\vdash X \otimes Y, (X^{\perp} \parr Y^{\perp}) \parr Z, Z^{\perp}$$

Règles :

$$\frac{}{\vdash X^{\perp}, X}\ (ax)$$

Axiome, règle de base « $X \implies X$ »

$$\frac{\vdash A, \Gamma \qquad \vdash B, \Delta}{\vdash A \otimes B, \Gamma, \Delta}\ (\otimes)$$

« Si $\Gamma \implies A$ et $\Delta \implies B$, alors Γ et $\Delta \implies A \otimes B$ »

$$\frac{\vdash A, B, \Gamma}{\vdash A \parr B, \Gamma}\ (\parr)$$

Remplace un ou « , » par un ou « $\parr$ »

Vers les réseaux

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \quad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$

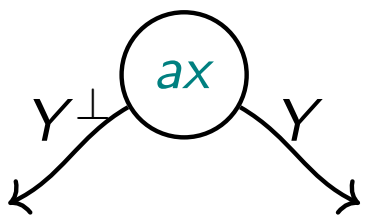
Vers les réseaux

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \quad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \quad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \quad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \quad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp \parr (X \otimes Y^\perp), Y \otimes Z^\perp, Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$

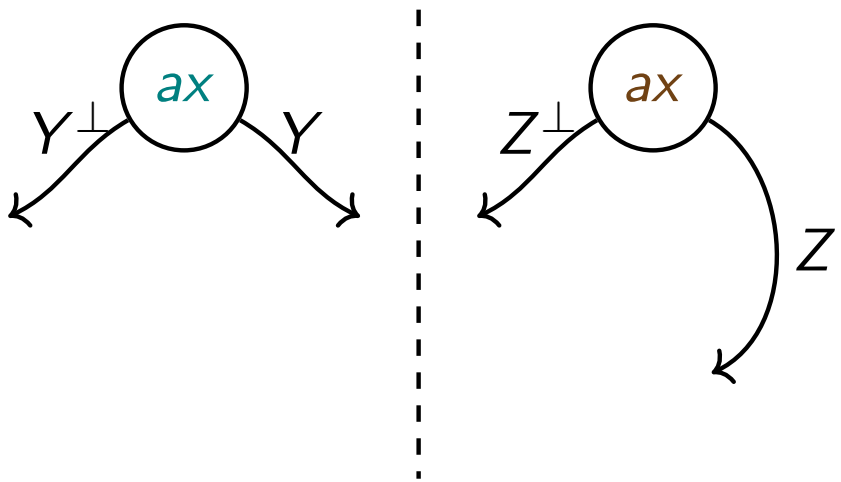
Vers les réseaux

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \quad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \quad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$



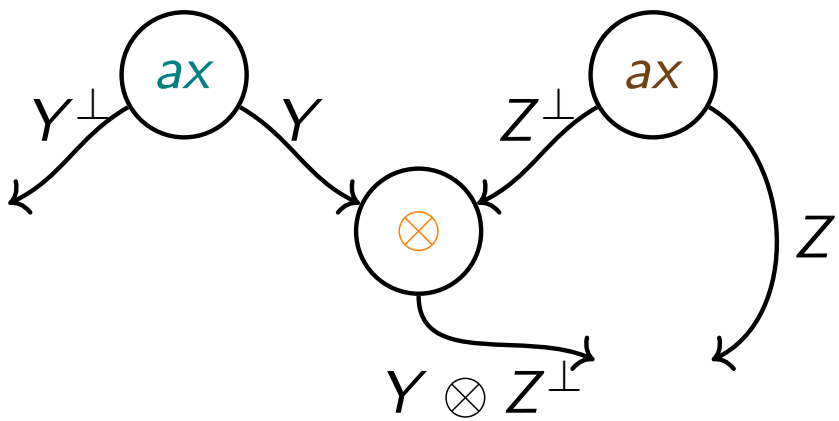
Vers les réseaux

$$\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \qquad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$



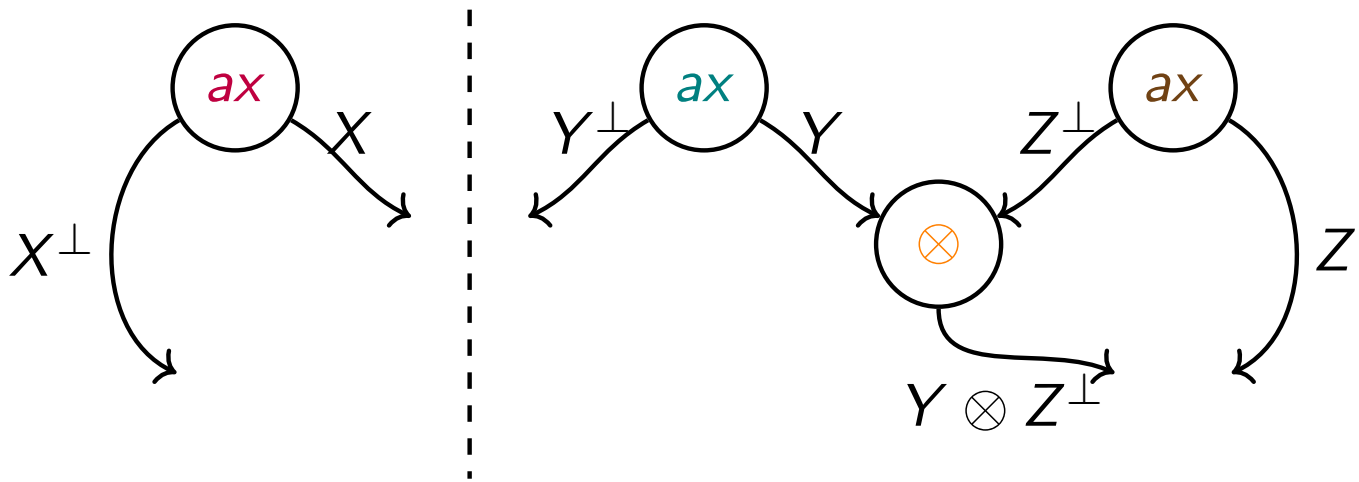
Vers les réseaux

$$\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \qquad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$



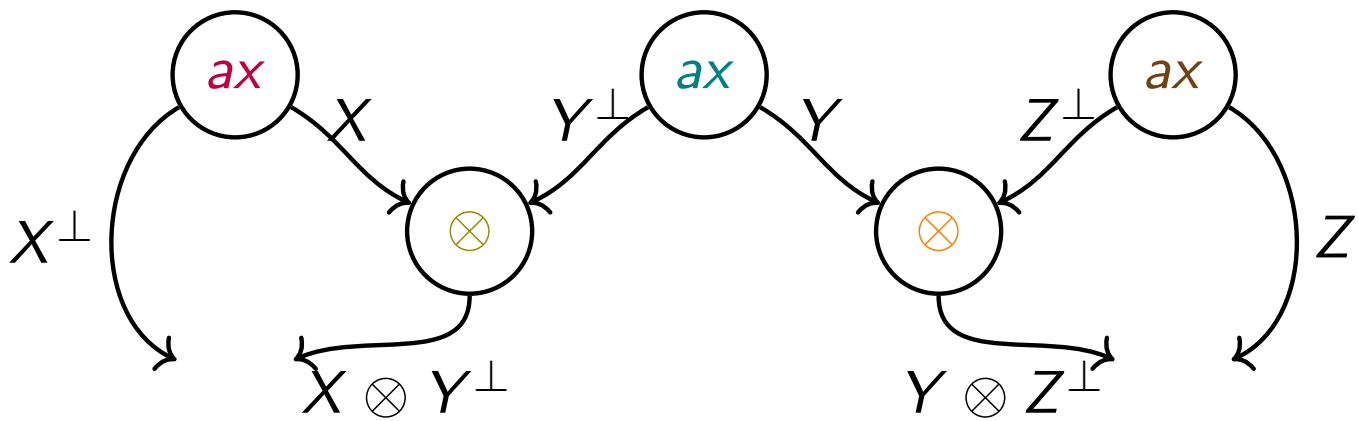
Vers les réseaux

$$\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \qquad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\dfrac{\dfrac{\dfrac{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)}}$$



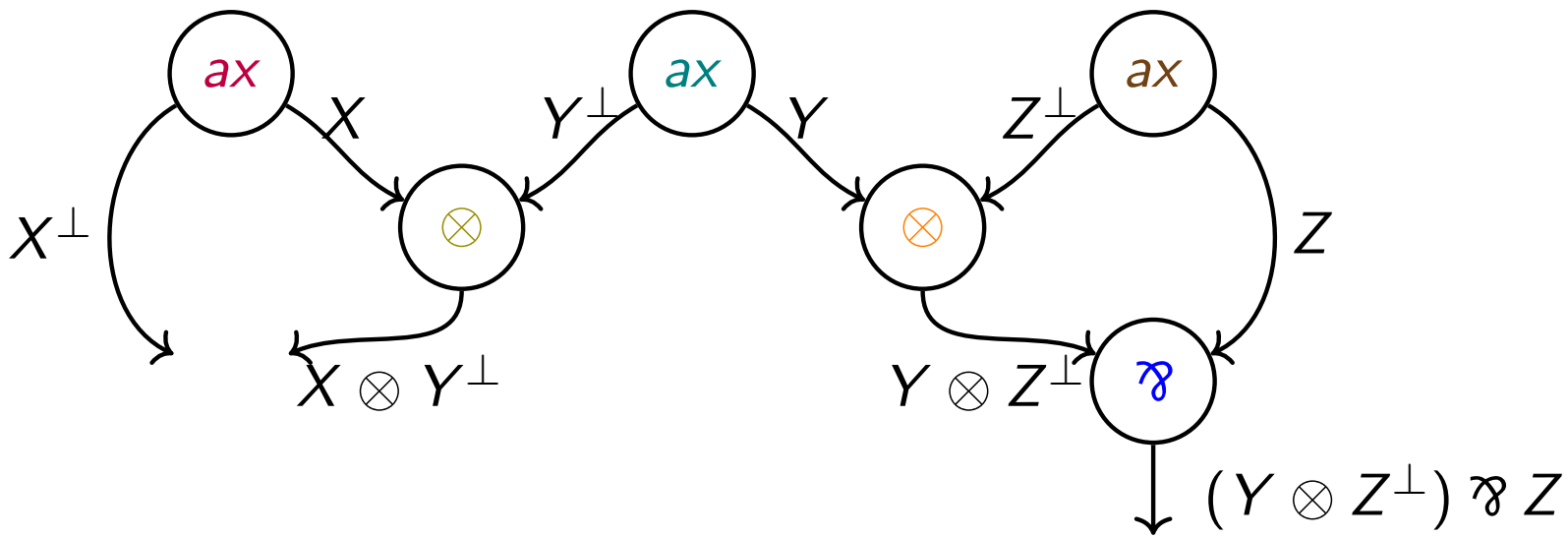
Vers les réseaux

$$\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \qquad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\dfrac{\dfrac{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)}\,(\otimes)$$



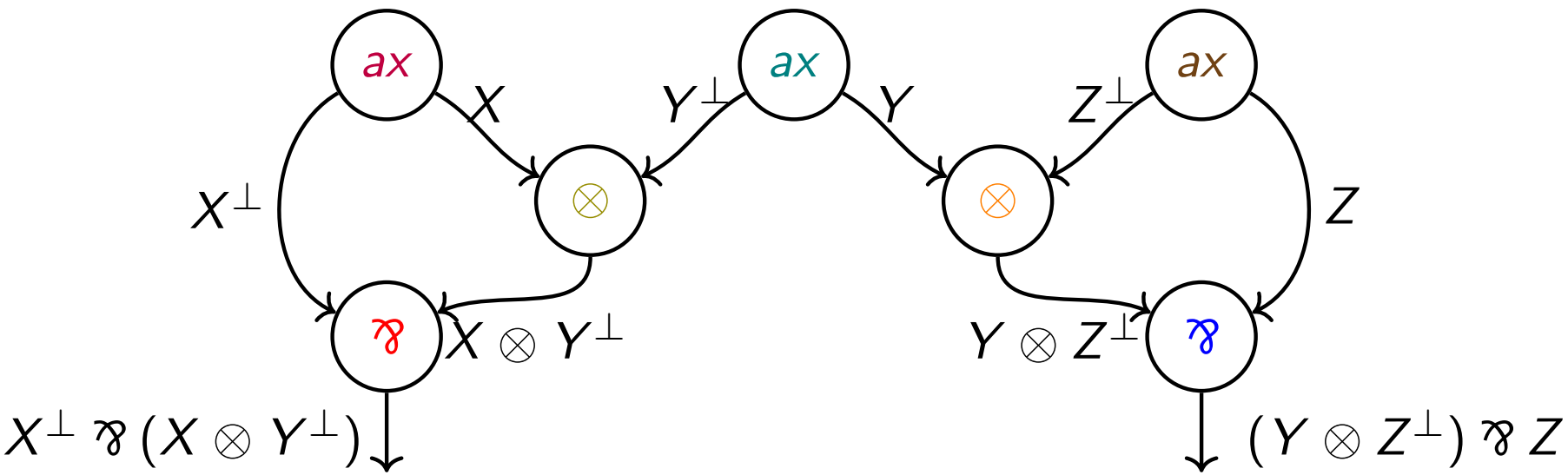
Vers les réseaux

$$\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \qquad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$



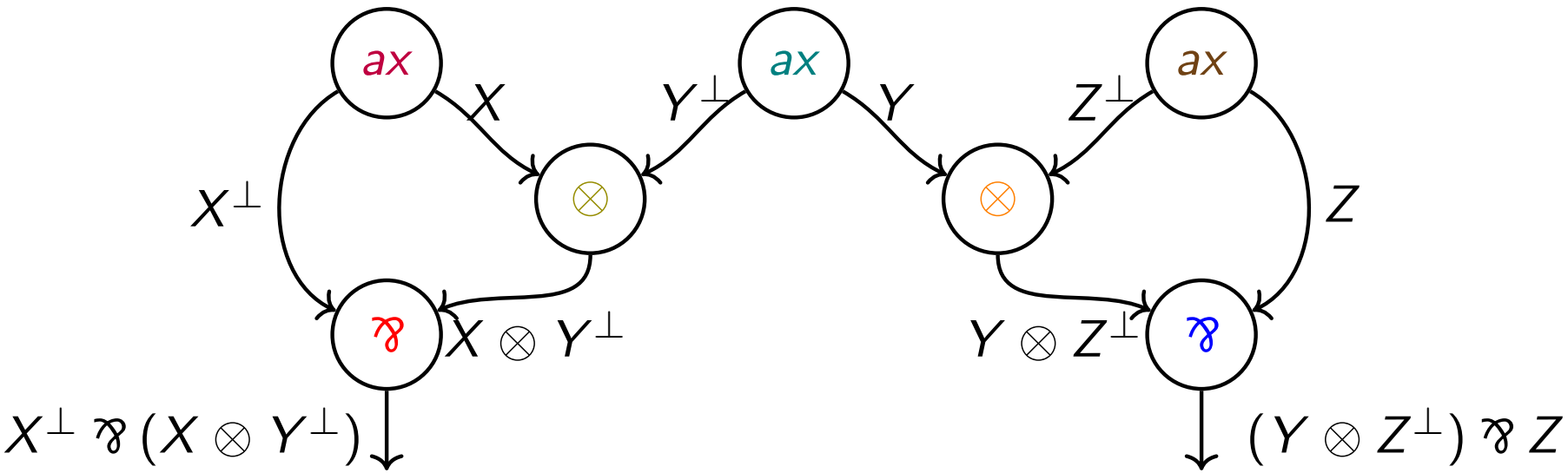
Vers les réseaux

$$\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \qquad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, (Y \otimes Z^\perp) \parr Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$



Vers les réseaux

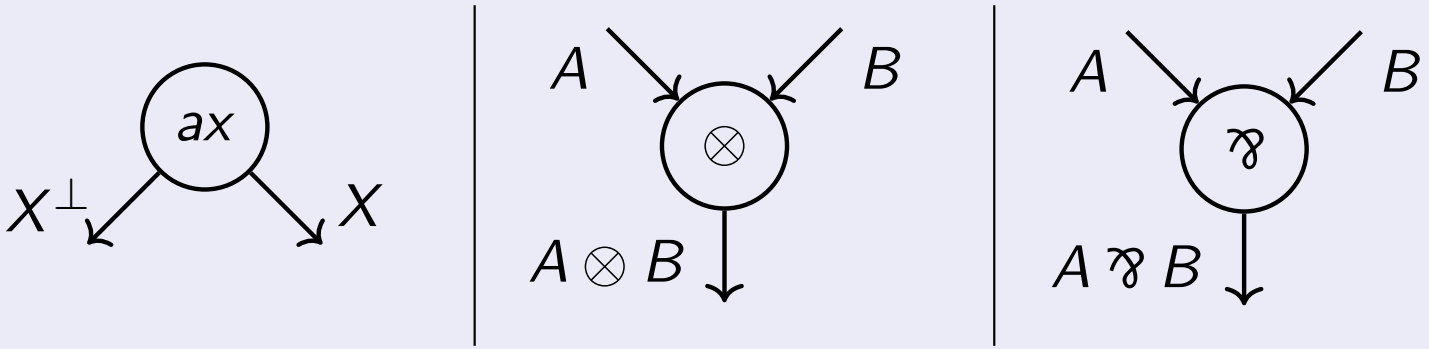
$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash X^\perp, X}\,(ax) \qquad \dfrac{\dfrac{}{\vdash Y^\perp, Y}\,(ax) \qquad \dfrac{}{\vdash Z^\perp, Z}\,(ax)}{\vdash Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp, X \otimes Y^\perp, Y \otimes Z^\perp, Z}\,(\otimes)}{\vdash X^\perp \parr (X \otimes Y^\perp), Y \otimes Z^\perp, Z}\,(\parr)}{\vdash X^\perp \parr (X \otimes Y^\perp), (Y \otimes Z^\perp) \parr Z}\,(\parr)$$

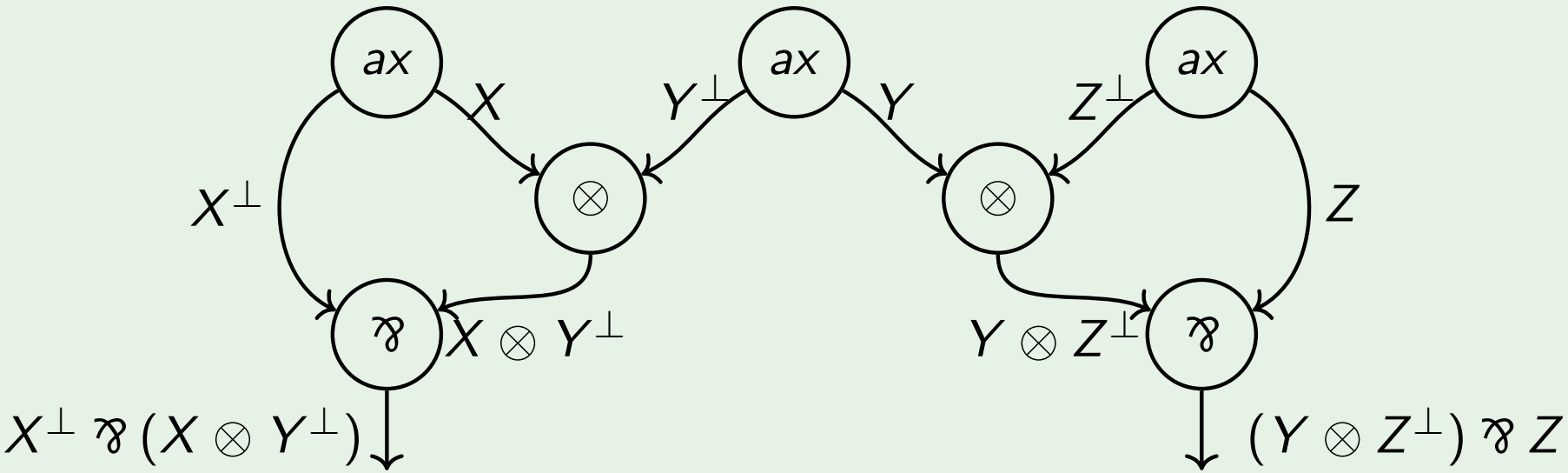


Structure de preuve

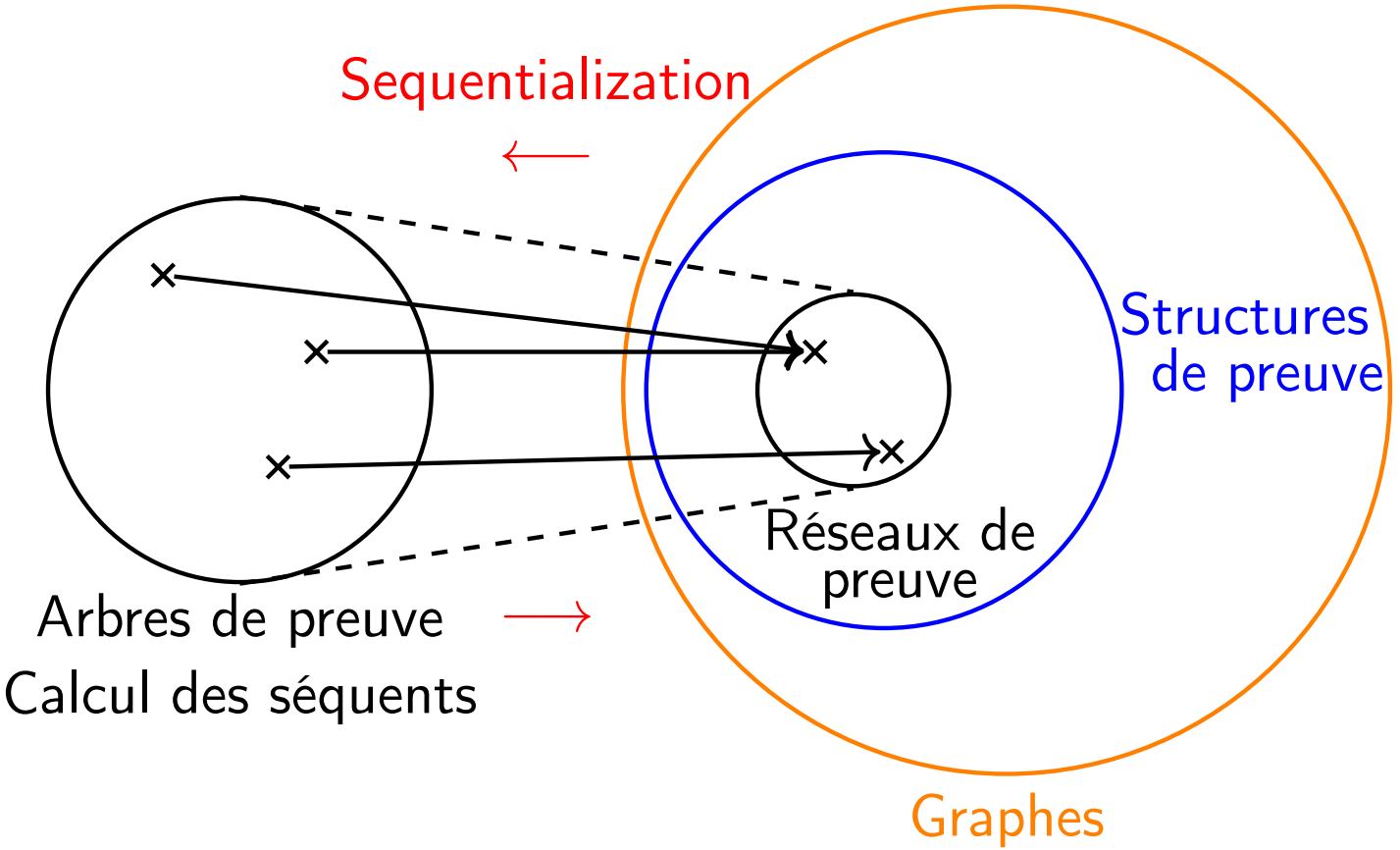
Definition

Multigraphe orienté, partiel avec étiquettes sur les sommets $\rightarrow ax/\otimes/\parr$
sur les arêtes $\rightarrow$ formules





Résumé


Sequentialization
Structures de preuve
Réseaux de preuve
Arbres de preuve
Calcul des séquents
Graphes

Plan

- Logique Linéaire Multiplicative
  - Calcul des séquents
  - Réseaux de preuve

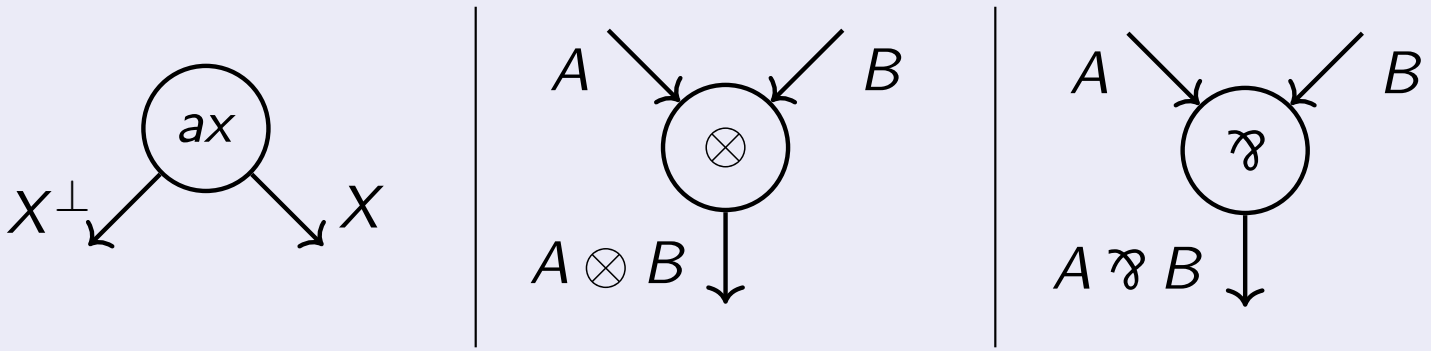
- **Formalisation des réseaux de preuve**
  - Structures de preuve
  - Temps de calcul

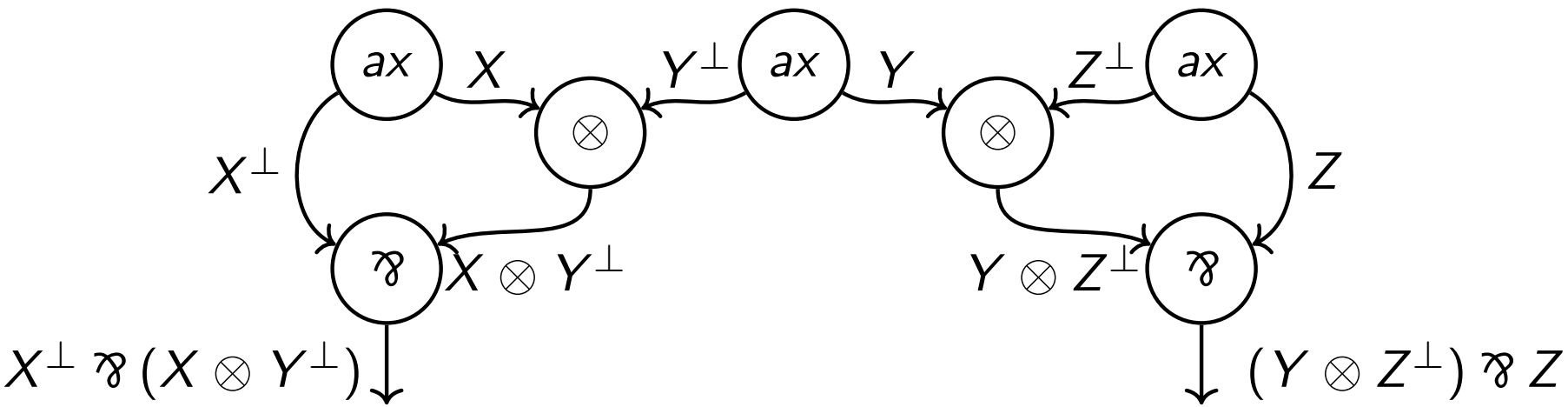
Choix et formalisation des structures de preuve

Formaliser cette définition = choix crucial influençant toute le reste

Definition

Multigraphe orienté, partiel avec étiquettes sur les sommets $\rightarrow ax / \otimes / \parr$
sur les arêtes $\rightarrow$ formules



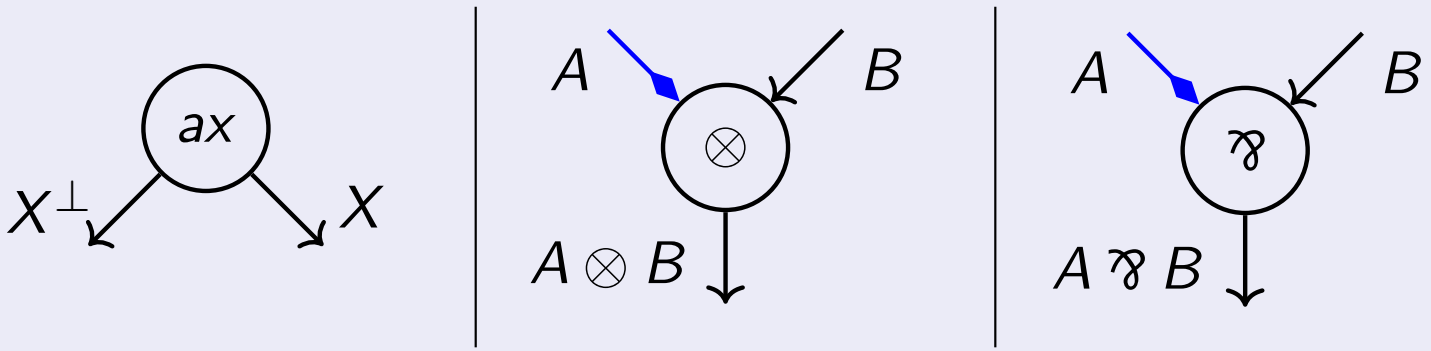


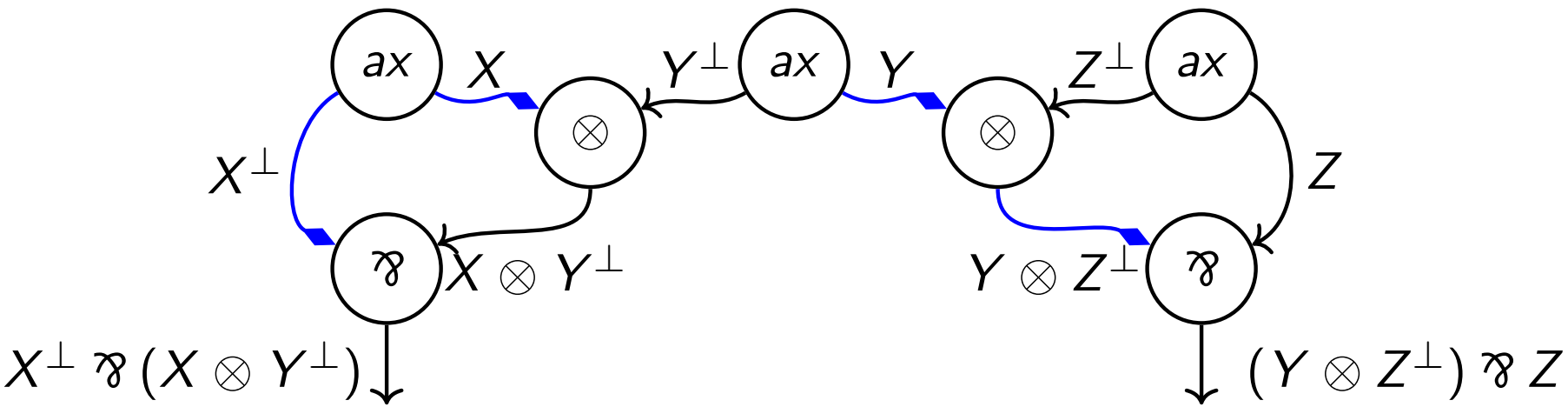
Choix et formalisation des structures de preuve

Formaliser cette définition = choix crucial influençant toute le reste

Definition

Multigraphe orienté, partiel avec étiquettes sur les sommets $\rightarrow ax / \otimes / \parr$
sur les arêtes $\rightarrow$ formules





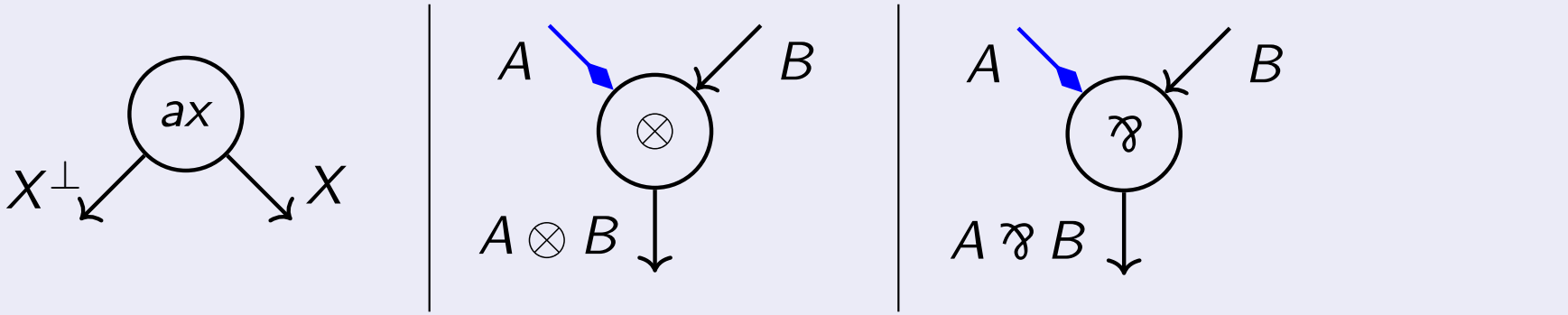
Choix et formalisation des structures de preuve

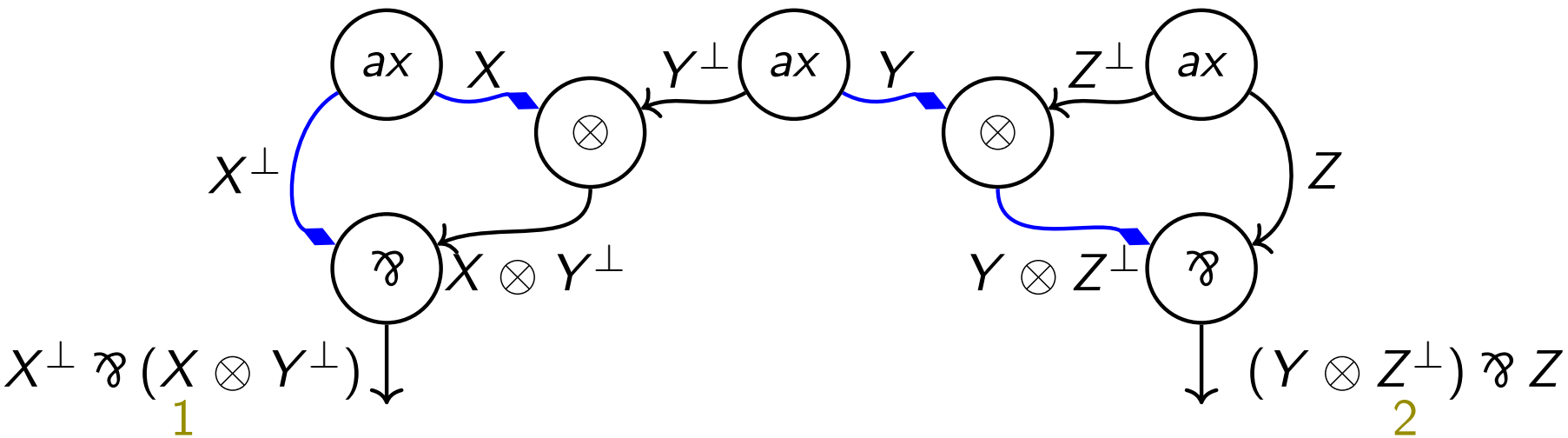
Formaliser cette définition = choix crucial influençant toute le reste

Definition

Multigraphe orienté, partiel avec étiquettes sur les sommets $\rightarrow ax / \otimes / \parr$
sur les arêtes $\rightarrow$ formules

et un ordre sur les conclusions





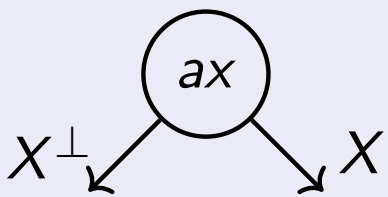
Choix et formalisation des structures de preuve

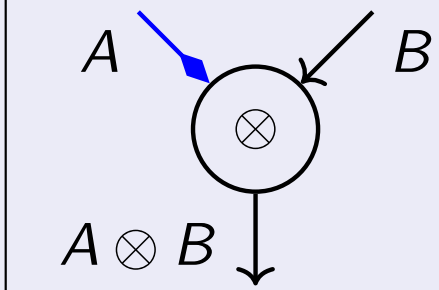
Formaliser cette définition = choix crucial influençant toute le reste

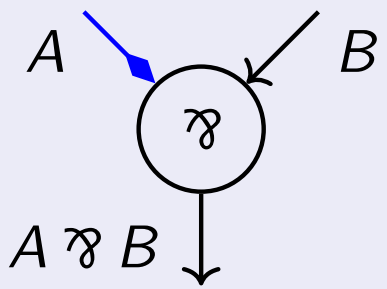
Definition

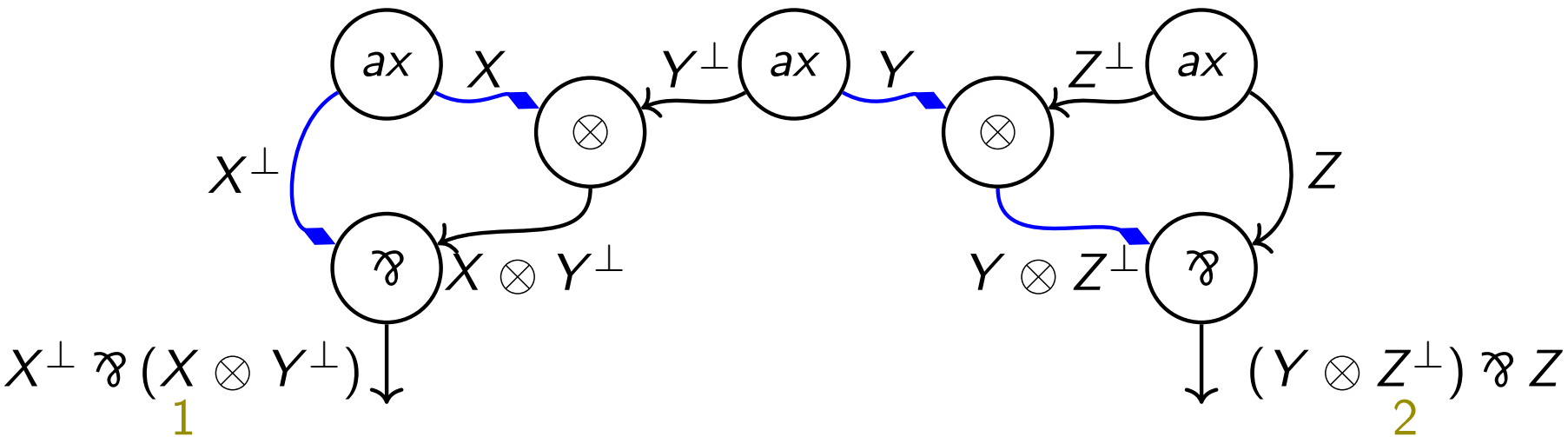
Multigraphe orienté, partiel avec étiquettes sur les sommets $\rightarrow ax / \otimes / \parr$
sur les arêtes $\rightarrow$ formules

et un ordre sur les conclusions









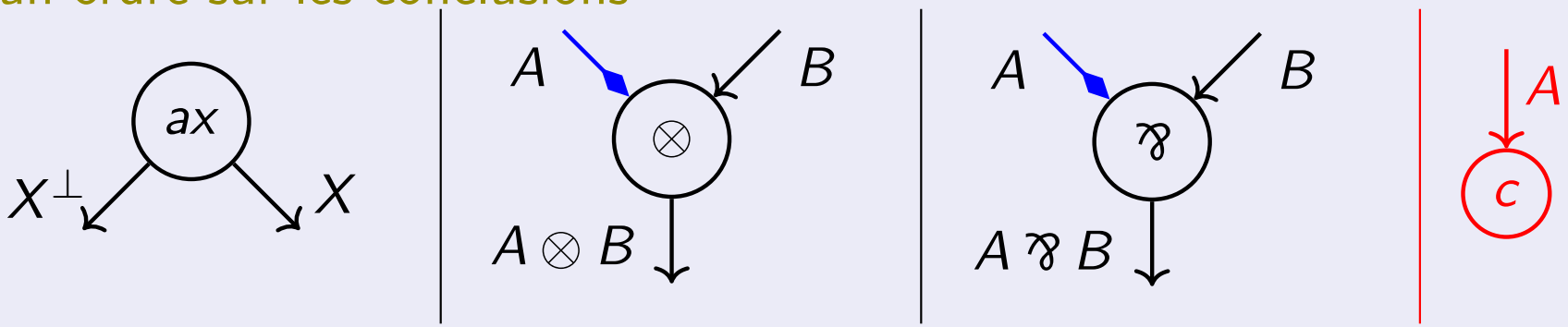
Choix et formalisation des structures de preuve

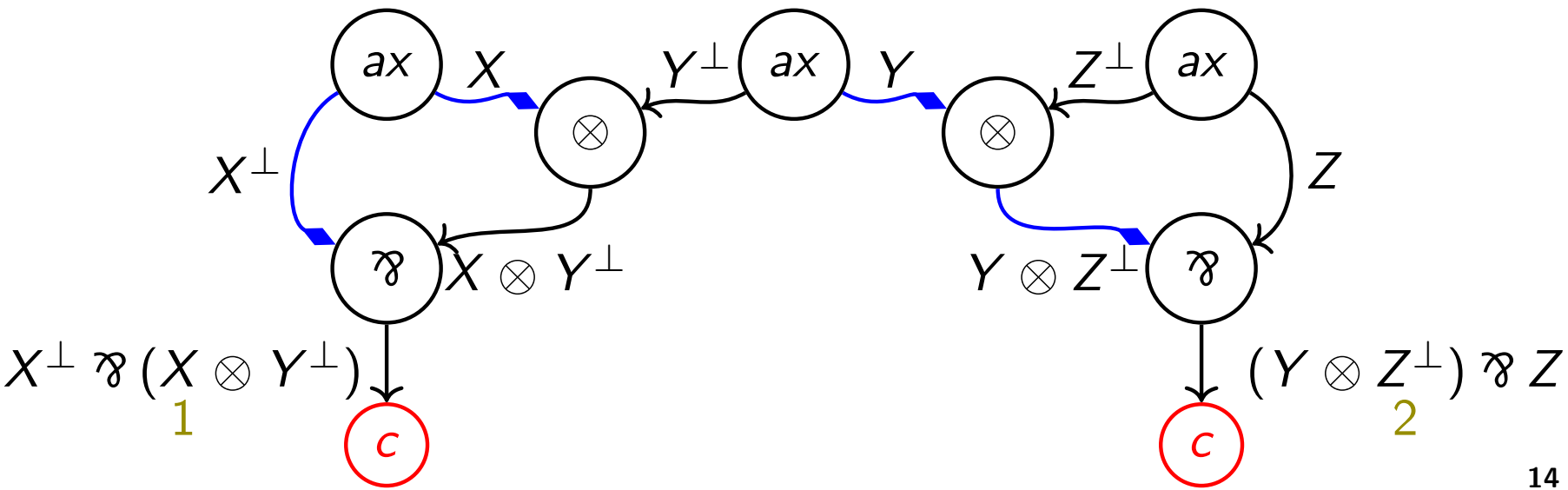
Formaliser cette définition = choix crucial influençant toute le reste

Definition

Multigraphe orienté, ~~partiel~~ avec étiquettes sur les sommets $\rightarrow ax / \otimes / \parr / c$
sur les arêtes $\rightarrow$ formules

et un ordre sur les conclusions





Choix et formalisation des structures de preuve

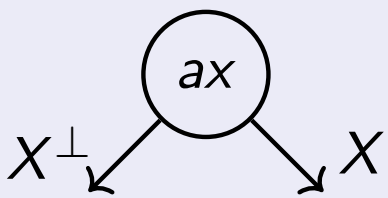
Formaliser cette définition = choix crucial influençant toute le reste

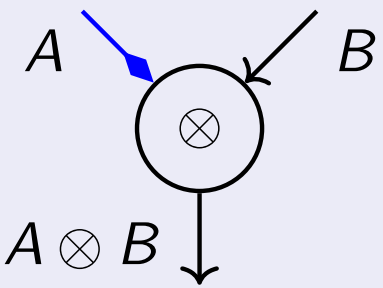
Definition

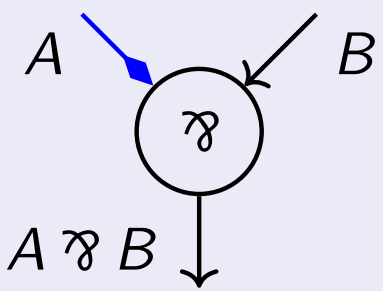
Multigraphe orienté, ~~partiel~~ avec étiquettes sur les sommets $\rightarrow ax / \otimes / \parr / c$

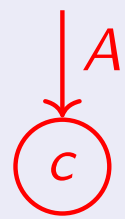
sur les arêtes $\rightarrow$ formules

et un ordre sur les conclusions









Autres variations :

- arêtes bidirectionnelles plutôt que sommets *ax*
- hypergraphe plutôt que multigraphe
- différents critères de correction
- ...

Implémentation des graphes manipulés

Traduction directe :

```
Record graph_data : Type :=
  Graph_data {
    graph_of :> graph rule formula;
     left : { v : vertex graph_of | vlabel v == ⊗ || vlabel v == ⅋ }
          → edge graph_of;
     right : { v : vertex graph_of | vlabel v == ⊗ || vlabel v == ⅋ }
          → edge graph_of;
     order : { v : vertex graph_of | vlabel v == c } →
          'I_#|{ v : vertex graph_of | vlabel v == c }|;
  }.
```

Implémentation des graphes manipulés

Traduction directe :

```
Record graph_data : Type :=
  Graph_data {
    graph_of :> graph rule formula;
     left : { v : vertex graph_of | vlabel v == ⊗ || vlabel v == ⅋ }
          → edge graph_of;
     right : { v : vertex graph_of | vlabel v == ⊗ || vlabel v == ⅋ }
          → edge graph_of;
     order : { v : vertex graph_of | vlabel v == c } →
          'I_#|{ v : vertex graph_of | vlabel v == c }|;
  }.
```

⟶ Types dépendants rapidement trop complexes

Exemple

Pour définir le graphe résultant de l'ajout d'un sommet, il faut prouver que chaque ⊗ avant l'ajout est toujours un ⊗ après, etc.

Implémentation des graphes manipulés bis

Sans les types dépendants :

```
Record graph_data : Type :=
  Graph_data {
    graph_of :> graph rule formula;
    left : vertex graph_of → edge graph_of;
    right : vertex graph_of → edge graph_of;
    order : vertex graph_of → int;
  }.
```

Implémentation des graphes manipulés bis

Sans les types dépendants :

```
Record graph_data : Type :=
  Graph_data {
    graph_of :> graph rule formula;
    left : vertex graph_of → edge graph_of;
    right : vertex graph_of → edge graph_of;
    order : vertex graph_of → int;
  }.
```

⟶ Donner des valeurs arbitraires pour les arguments non intéressants était parfois la plus longue partie de certaines définitions / preuves !

Implémentation des graphes manipulés bis

Sans les types dépendants :

```
Record graph_data : Type :=
  Graph_data {
    graph_of :> graph rule formula;
    left : vertex graph_of → option edge graph_of;
    right : vertex graph_of → option edge graph_of;
    order : vertex graph_of → option int;
  }.
```

⟶ Donner des valeurs arbitraires pour les arguments non intéressants était parfois la plus longue partie de certaines définitions / preuves !

⟶ Mettre des types options introduit un pattern matching sans intérêt absolument partout

Implémentation des graphes manipulés ter

Solution retenue :

```
Record graph_data : Type :=
  Graph_data {
    graph_of :> graph rule formula;
    left : edge graph_of → bool;
    order : seq (vertex graph_of);
  }.
```

- Un booléen pour indiquer si une arête est gauche ou non
- Liste de sommets pour l'ordre des conclusions
- Puis des propriétés à vérifier :
  - pas deux arêtes gauches pour un même sommet
  - la liste contient exactement les sommets c
  - ...

Implémentation des graphes manipulés ter

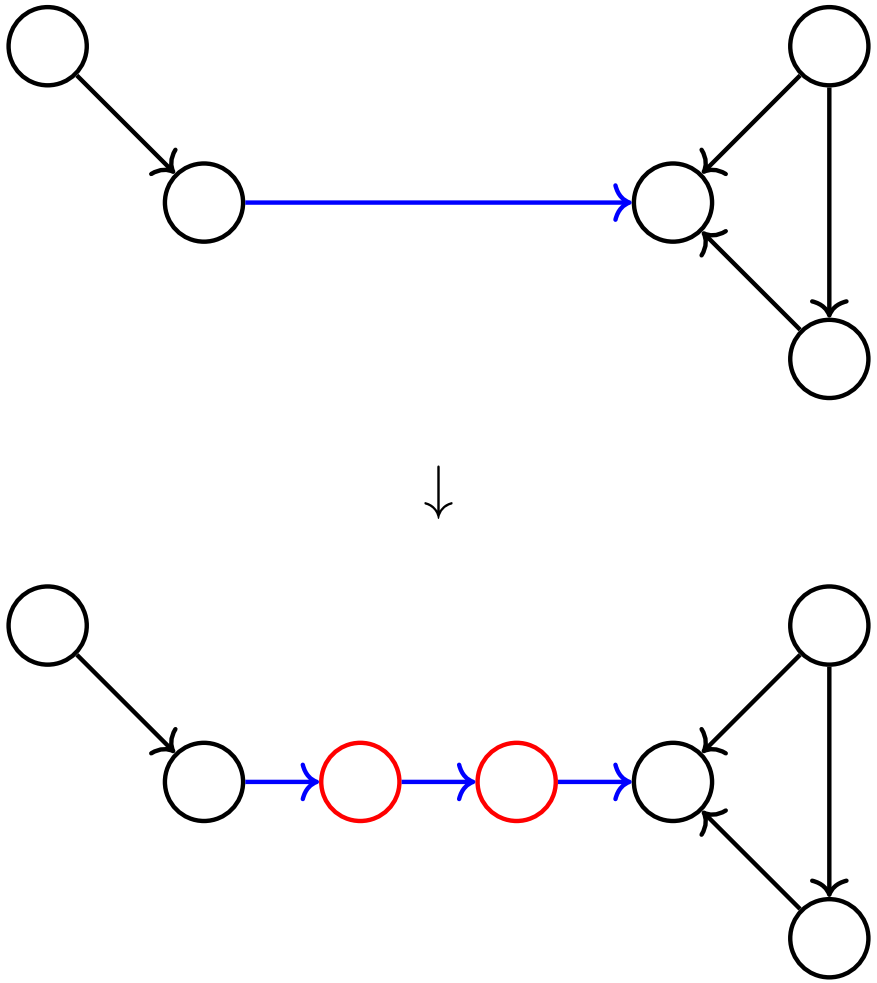
Solution retenue :

```
Record graph_data : Type :=
  Graph_data {
    graph_of :> graph rule formula;
    left : edge graph_of → bool;
    order : seq (vertex graph_of);
  }.
```

- Un booléen pour indiquer si une arête est gauche ou non
- Liste de sommets pour l'ordre des conclusions
- Puis des propriétés à vérifier :
  - pas deux arêtes gauches pour un même sommet
  - la liste contient exactement les sommets c
  - ...

$\longrightarrow$ Retire les difficultés des définitions et les place dans les preuves

Une opération simple . . .



. . . avec un temps de calcul trop long . . .

```
Definition extend_edge_graph (G : graph unit unit) (e : edge G) :
graph unit unit :=
  remove_edges [set e : edge G] ⊹ tt ⊹ [inl (source e), tt, inr tt] ⊹ [inr
tt, tt, inl (target e)].

Definition new_graph (G : graph unit unit) (e : edge G) :=
(@extend_edge_graph (@extend_edge_graph G e) None).

Fail Time Definition transport_to_new (G : graph unit unit) (e : edge
G) :
edge G → edge (new_graph e) :=
  fun a ⇒ if @boolP (a \notin [set e]) is AltTrue p1 then
    if @boolP (Some (Some (inl (Sub a p1)))) \notin [set None]) is
AltTrue p2 then
      Some (Some (inl (Sub (Some (Some (inl (Sub a p1)))) p2)))
    else None else None.
```

. . . ou avec trop d'annotations

```
Time Definition transport_to_new (G : graph unit unit) (e : edge G) :
edge G → edge (new_graph e) :=
  fun a ⇒ if @boolP (a \notin [set e]) is AltTrue p1 then
    if @boolP ((Some (Some (inl (Sub a p1 : edge (remove_edges [set
e : edge G])))) : edge (@extend_edge_graph G e)) \notin [set None]) is
AltTrue p2 then
      Some (Some (inl (Sub (Some (Some (inl (Sub a p1 : edge
(remove_edges [set e : edge G])))) : edge (@extend_edge_graph G e)))
p2 : edge (remove_edges [set None : edge (@extend_edge_graph G
e)])))))
    else None else None.
```

Conclusion

- Définition des réseaux de preuve.
- Fonction des arbres du calcul des séquents vers les graphes . . .
- . . . et une preuve que les graphes obtenus sont des réseaux.
- Une preuve du théorème de séquentialisation : un réseau est (isomorphe à) l'image d'un arbre du calcul des séquents.
- Définition de l'élimination des coupures (un système de réécriture) . . .
- . . . et une preuve que l'appliquer préserve être un réseau . . .
- . . . et une preuve que ce système termine.
- Résultats intermédiaires sur les graphes, pas dans **GraphTheory** (théorie des chemins non orientés, etc).

Perspectives

- Expansion des axiomes (avec confluence et terminaison).
- Liens entre calcul des séquents et réseaux sur l'élimination des coupures et l'expansion des axiomes.
- Les réseaux quotientent les arbres des séquents par les commutations de règles.
- Des résultats utilisants les réseaux : caractérisation des isomorphismes par Balat et Di Cosmo [BD99], ...
- Réseaux pour des systèmes plus larges : MELL et MALL.

Merci !

References I

[BD99] Vincent Balat and Roberto Di Cosmo. "A Linear Logical View of Linear Type Isomorphisms". In: *Computer Science Logic*. Ed. by Jörg Flum and Mario Rodríguez-Artalejo. Vol. 1683. Lecture Notes in Computer Science. Springer, 1999, pp. 250–265.

[CL] Pierre Courtieu and Olivier Laurent. *ill_narratives*. Coq library. URL: https://github.com/Matafou/ill_narratives.

References II

[CLR17] Kaustuv Chaudhuri, Leonardo Lima, and Giselle Reis. "Formalized Meta-Theory of Sequent Calculi for Substructural Logics". In: *Electronic Notes in Theoretical Computer Science* 332 (June 2017). LSFA 2016 - 11th Workshop on Logical and Semantic Frameworks with Applications (LSFA), pp. 57–73. DOI: 10.1016/j.entcs.2017.04.005. URL: https://www.sciencedirect.com/science/article/pii/S1571066117300154.

References III

[CLR19] Kaustuv Chaudhuri, Leonardo Lima, and Giselle Reis. "Formalized meta-theory of sequent calculi for linear logics". In: *Theoretical Computer Science* 781 (2019). Code source at https://github.com/meta-logic/abella-reasoning, pp. 24–38. DOI: https://doi.org/10.1016/j.tcs.2019.02.023. URL: https://www.sciencedirect.com/science/article/pii/S030439751930129X.

[Gro95] Philippe de Groote. "Linear logic with isabelle: Pruning the proof search tree". In: *Theorem Proving with Analytic Tableaux and Related Methods*. Ed. by Peter Baumgartner, Reiner Hähnle, and Joachim Possega. Springer, Sept. 1995, pp. 263–277. DOI: 10.1007/3-540-59338-1_41. URL: https://link.springer.com/chapter/10.1007/3-540-59338-1_41.

References IV

[KP95] Sara Kalvala and Valeria de Paiva. "Mechanizing Linear Logic in Isabelle". In: *Isabelle Users Workshop*. Sept. 1995. URL: https://www.cl.cam.ac.uk/~lp15/papers/Workshop/papers/kalvala-linear.pdf.

[Lau17] Olivier Laurent. *Yalla: Yet Another deep embedding of Linear Logic in Coq*. Coq library. July 2017. URL: https://perso.ens-lyon.fr/olivier.laurent/yalla/.

[PC] Pierre-Marie Pédrot and Guillaume Claret. *ll_coq*. Coq library. URL: https://github.com/ppedrot/ll-coq.

References V

[PW99] James Power and Caroline Webster. "Working with Linear Logic in Coq". In: *Theorem Proving in Higher Order Logics: Emerging Trends*. An implementation is available at `https://github.com/ComputerAidedLL/PowerWebster_ILL`. Sept. 1999. URL: `http://www-sop.inria.fr/croap/TPHOLs99/proceeding.html`.

[Sad03] Mehrnoosh Sadrzadeh. "Modal Linear Logic in Higher Order Logic, an experiment in Coq". In: *Theorem Proving in Higher Order Logics (01/09/03)*. Ed. by D Basin and W Burkhart. Sept. 2003, pp. 75–93. URL: `https://eprints.soton.ac.uk/261814/`.

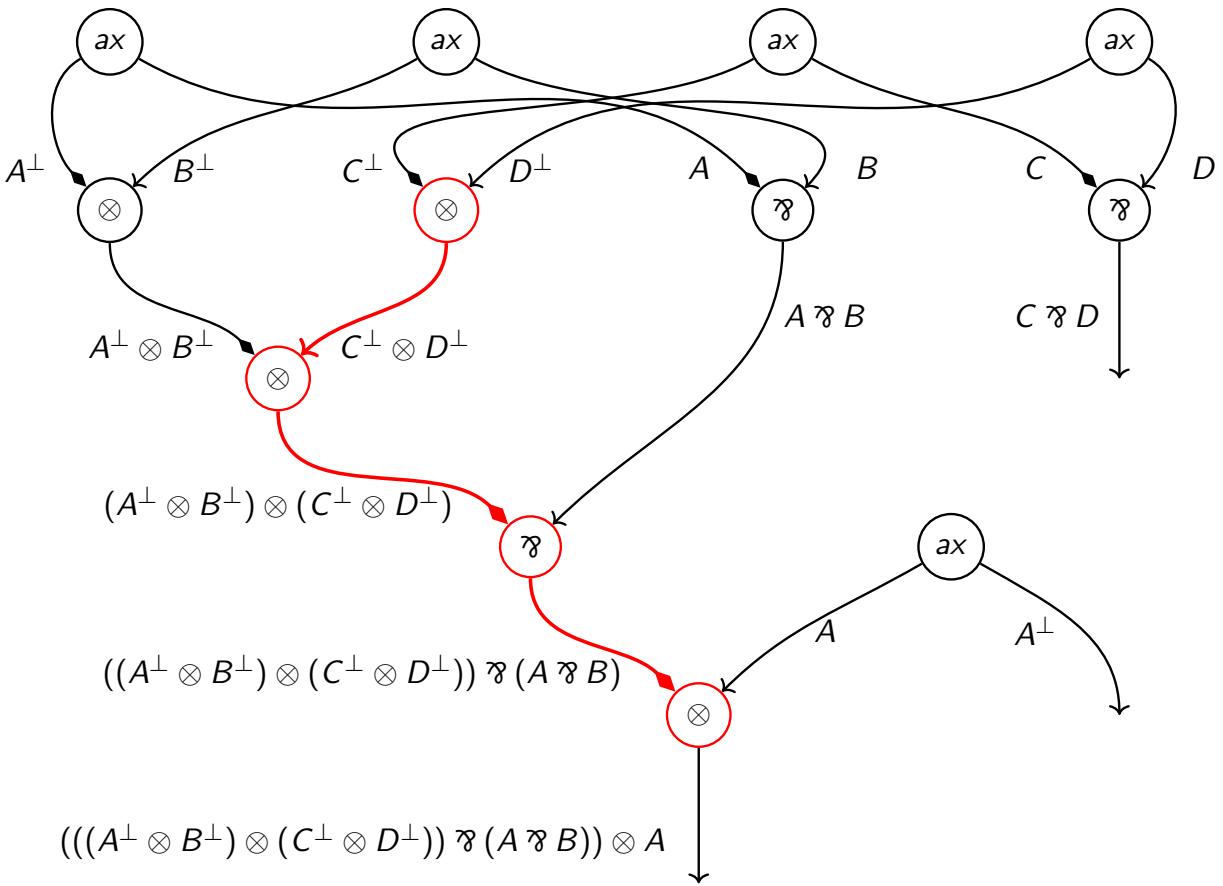
References VI

[Xav+18] Bruno Xavier, Carlos Olarte, Giselle Reis, and Vivek Nigam. "Mechanizing Focused Linear Logic in Coq". In: *12th Workshop on Logical and Semantic Frameworks with Applications (LSFA 2017)*. Ed. by Sandra Alves and Renata Wassermann. Vol. 338. Code source available at https://github.com/meta-logic/coq-ll. 2018, pp. 219–236. DOI: 10.1016/j.entcs.2018.10.014. URL: https://www.sciencedirect.com/science/article/pii/S157106611830080X.

Raisonnement informel en théorie des graphes

Chemin descendant

Un *chemin descendant* est un chemin maximal d'un sommet vers un sommet terminal (dont les arêtes sortantes sont des conclusions).



Raisonnement formel en théorie des graphes

Lemma

Une structure de preuve est un DAG (directed acyclic multigraph).

Lemma

La relation "être relié par un chemin" induite par un DAG est bien fondée.

Proposition

Pour un sommet (non étiqueté c) s dans une structure de preuve, il existe un chemin orienté de s vers t, avec t au dessus d'un sommet étiqueté c.

Critère de correction & Réseau de preuve

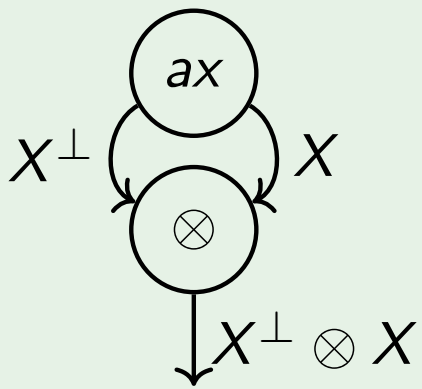
Graphe de correction

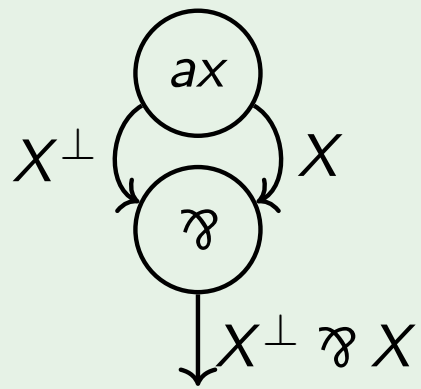
Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemples jouets





Critère de correction & Réseau de preuve

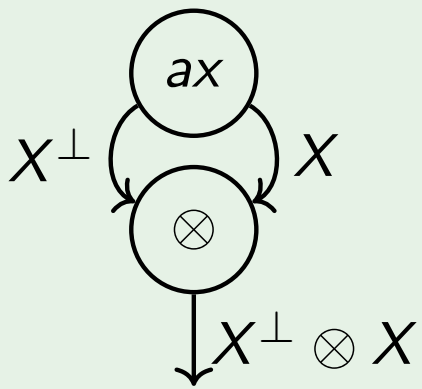
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

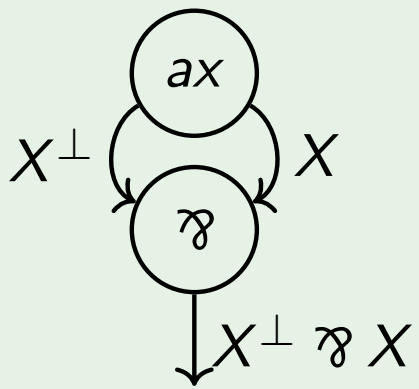
Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemples jouets



pas acyclique (mais connecté)

INCORRECTE



Critère de correction & Réseau de preuve

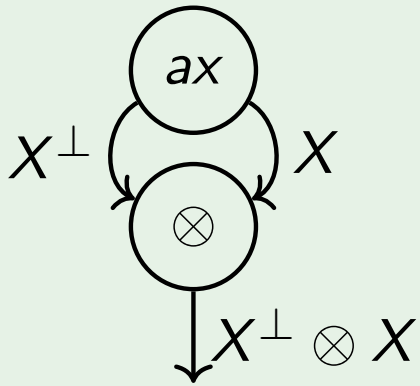
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

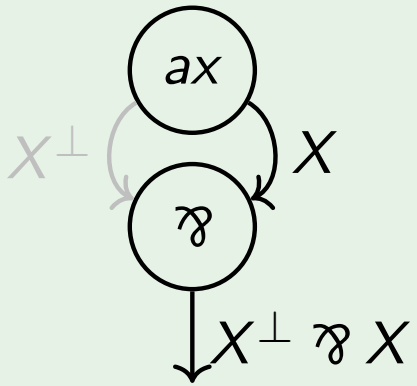
Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemples jouets



pas acyclique (mais connecté)

INCORRECTE



acyclique et connecté

Critère de correction & Réseau de preuve

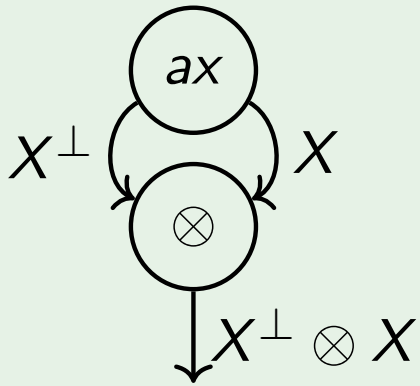
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

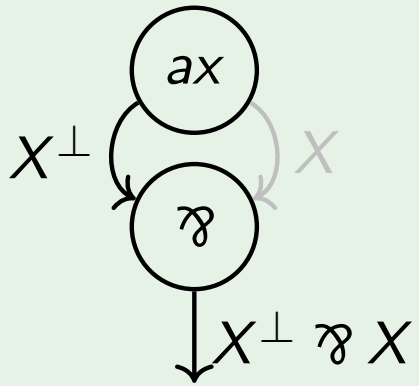
Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemples jouets



pas acyclique (mais connecté)

INCORRECTE



acyclique et connecté

CORRECTE

Critère de correction & Réseau de preuve

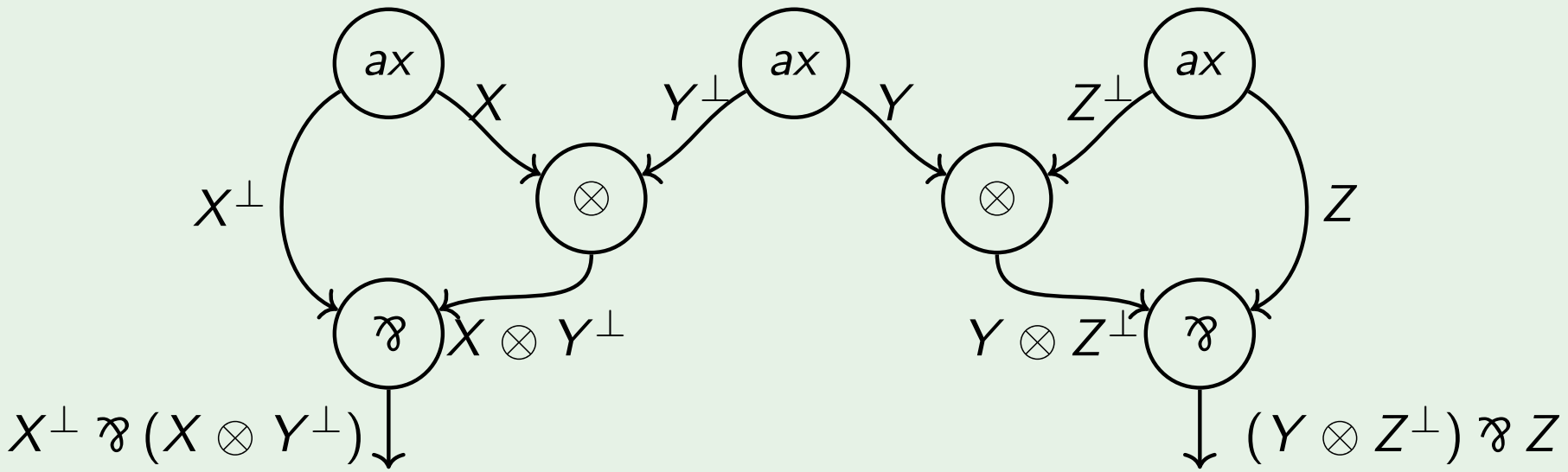
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemple précédent



Critère de correction & Réseau de preuve

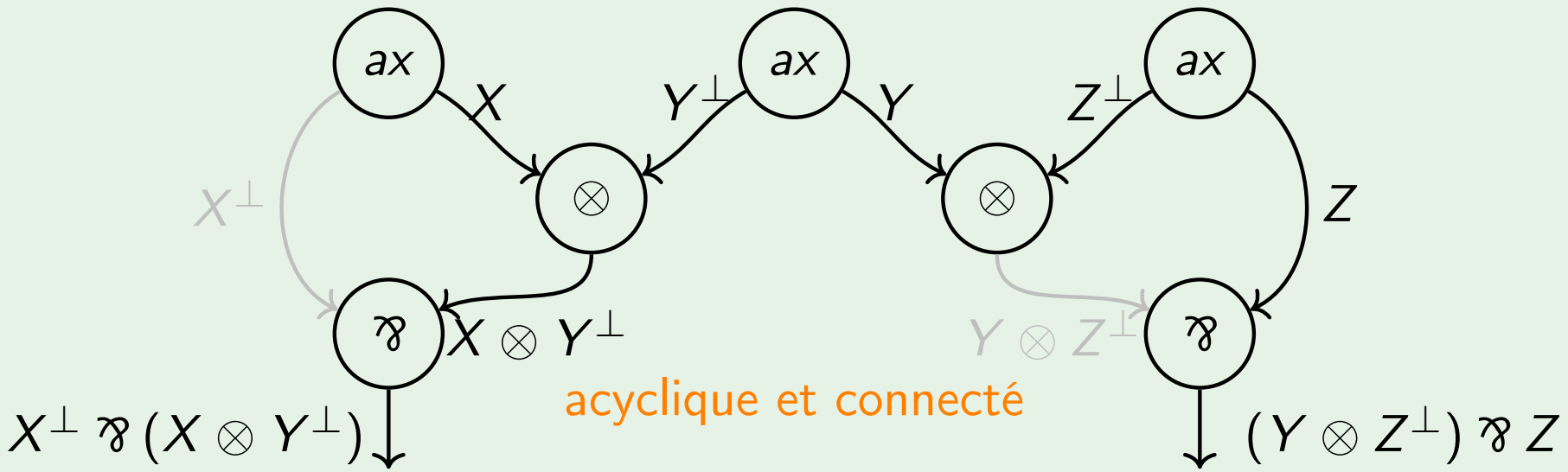
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemple précédent



Critère de correction & Réseau de preuve

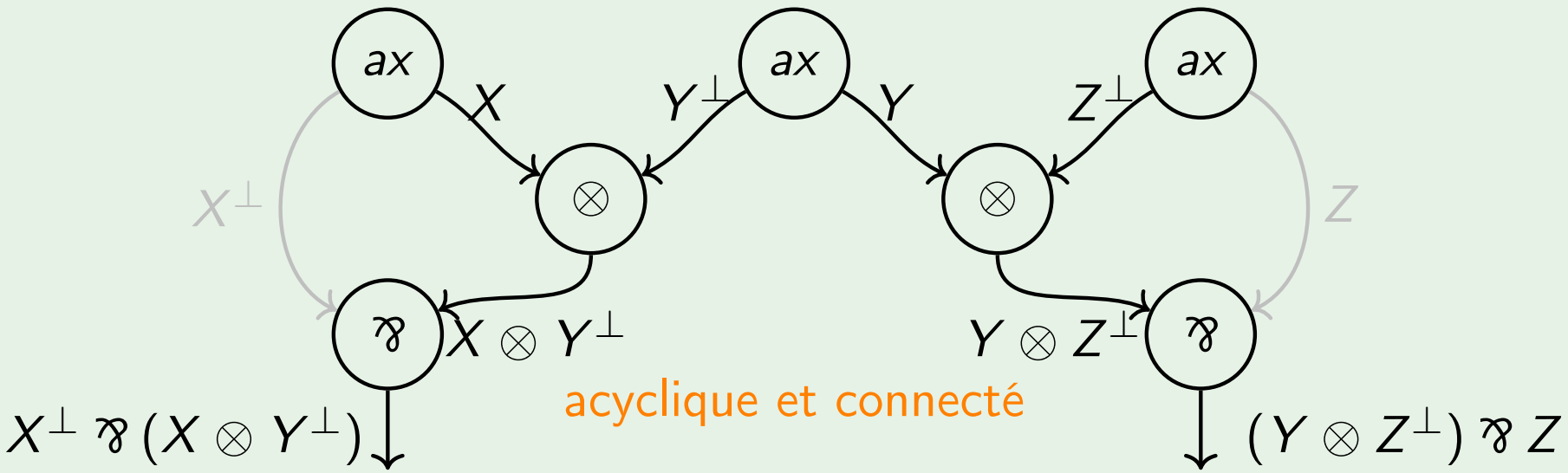
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemple précédent



Critère de correction & Réseau de preuve

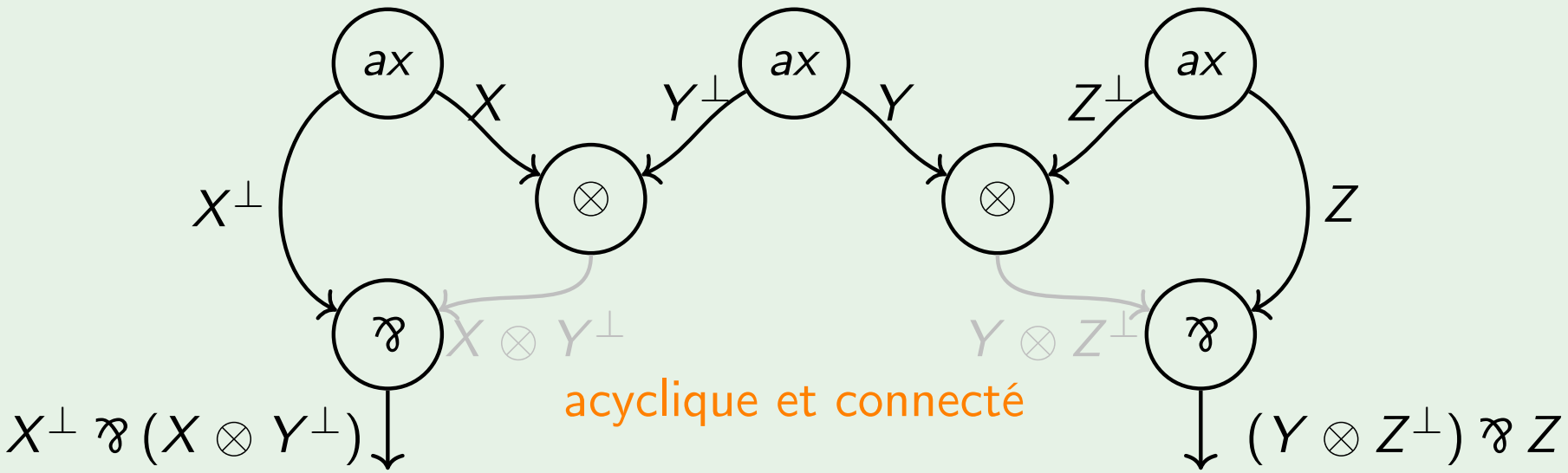
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemple précédent



Critère de correction & Réseau de preuve

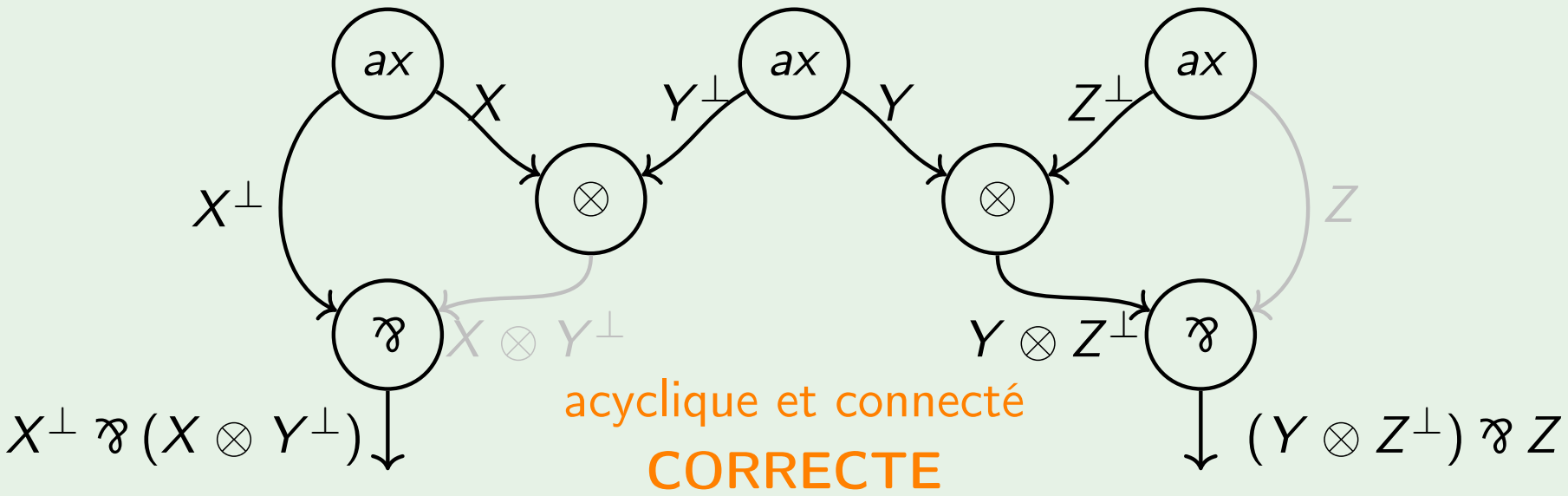
Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Exemple précédent



Critère de correction & Réseau de preuve

Graphe de correction

Dans une structure, retirer une arête entrante de chaque $\parr$.

Critère de correction

Une structure est *correcte*, et est nommée *réseau*, si **tous** ses graphes de correction sont **acycliques** et **connexes** (= des arbres).

Théorème de séquentialisation

$$\textit{correcte} \iff \textit{est l'image d'une preuve}$$

Critère de correction

Rappel

Graphe de correction : retirer une arête entrante de chaque $\curlyvee$
Correction : graphes de correction tous **acycliques** et **connexes**

Sur machine : horrible de manipuler les graphes de correction

$\longrightarrow$ si ajoute un sommet v à G, graphes de correction de $G + v$ **isomorphes** aux graphes de correction de G auxquels on ajoute v

$\longrightarrow$ formellement **pas les mêmes arêtes** entre G et ses graphes de correction, on a juste un morphisme des arêtes d'un graphe de correction vers G

Critère de correction sans graphes de correction

Idée : formuler la correction directement dans la structure de preuve

Acyclicité des graphes de correction
$\iff$ chaque cycle (non-orienté) de la structure utilise les 2 prémisses d'un même $\parr$

Critère de correction sans graphes de correction

Idée : formuler la correction directement dans la structure de preuve

Acyclicité des graphes de correction
$\iff$ chaque cycle (non-orienté) de la structure utilise les 2 prémisses d'un même $\parr$

Composantes connexes d'un graphe acyclique

acyclique $\implies \#cc = \#sommets - \#arêtes$

$\longrightarrow$ graphes de correction tous connexes ssi celui sans les arêtes gauches est connexe

Connexité des graphes de correction
$\iff$ toute paire de sommets de la structure est reliée par un chemin (non-orienté) qui n'utilise aucune arête gauche d'un $\parr$

Chemins « f-simples »

Étude de chemins non-orientés dans la structure elle-même avec :

- **Acyclicité** $\longrightarrow$ rendre des arêtes exclusives
- **Connexité** $\longrightarrow$ interdire des arêtes

$\Longrightarrow$ nouvelle notion de chemins non orientés

Chemins « f-simples »

Étude de chemins non-orientés dans la structure elle-même avec :

- **Acyclicité** $\longrightarrow$ rendre des arêtes exclusives
- **Connexité** $\longrightarrow$ interdire des arêtes

$\Longrightarrow$ nouvelle notion de chemins non orientés

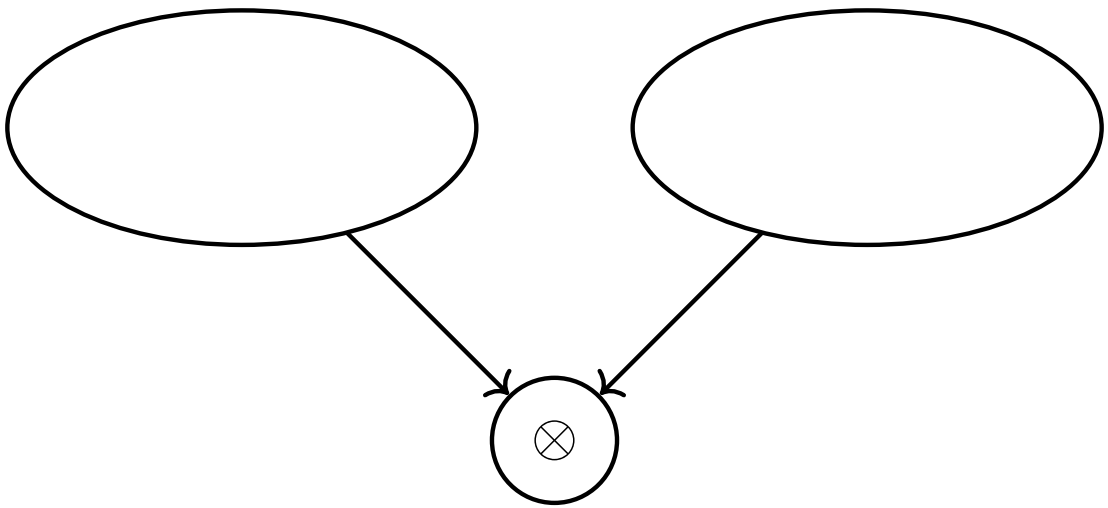
Chemins « f-simples »

Étant donné $f : E \longrightarrow I \cup \{\perp\}$

- injective sur les arêtes du chemin
- pas d'arête avec l'identifiant interdit $\perp$

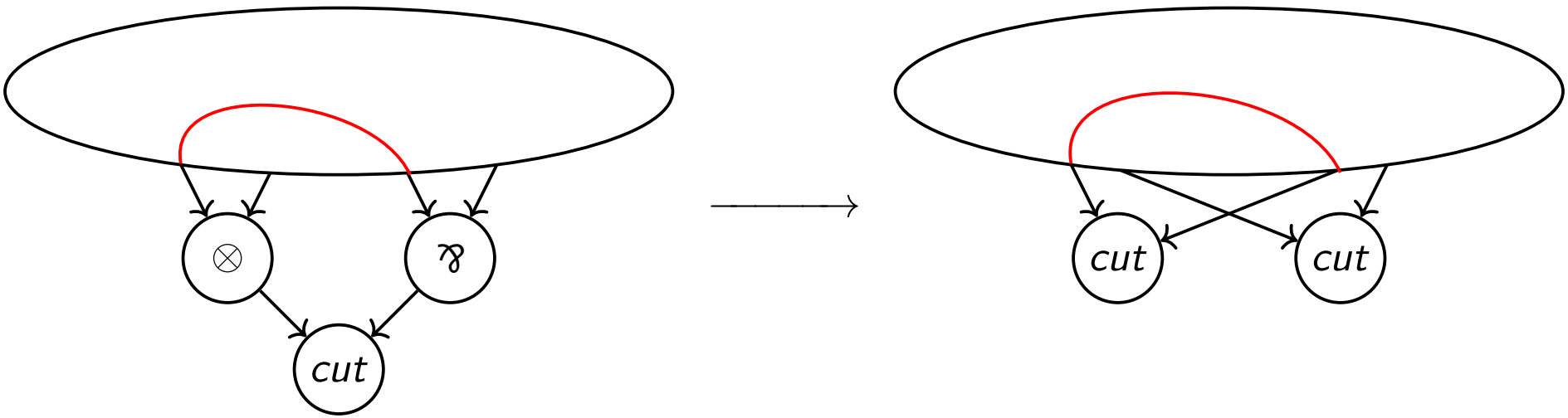
Exemples

Preuve de séquentialisation : trouve un sommet splittant comme sur papier, la suite plus complexe !



Exemples

Preuve de séquentialisation : trouve un sommet splittant comme sur papier, la suite plus complexe !



Transfert de chemins lorsqu'on ajoute ou retire un sommet, . . .

Strates de définition pour les réseaux Coq (1/2)

```
Notation base_graph := (graph (flat rule) (flat (formula × bool))).
Definition flabel {G : base_graph} (e : edge G) : formula :=
  fst (elabel e).
Definition llabel {G : base_graph} (e : edge G) : bool :=
  snd (elabel e).

Record graph_data : Type :=
  Graph_data {
    graph_of :> base_graph;
    order : seq (edge graph_of);
  }.

Definition sequent (G : graph_data) : seq formula :=
  [seq flabel e | e ← order G].
```

Strates de définition pour les réseaux Coq (2/2)

```
Record proof_structure : Type := Proof_structure {
    graph_data_of :> graph_data;
    p_deg : proper_degree graph_data_of;
    p_ax_cut : proper_ax_cut graph_data_of;
    p_tens_parr : proper_tens_parr graph_data_of;
    p_noleft : proper_noleft graph_data_of;
    p_order_full : proper_order_full graph_data_of;
    p_order_uniq : proper_order_uniq graph_data_of;
  }.

Definition proper_tens_parr (G : base_graph) :=
  ∀ (b : bool) (v : G), vlabel v = (if b then ⅋ else ⊗) →
  ∃ el er ec, el \in edges_at_in v ∧ llabel el ∧
    er \in edges_at_in v ∧ ¬llabel er ∧ ec \in edges_at_out v ∧
    flabel ec = (if b then ⅋ else ⊗) (flabel el) (flabel er).

Record proof_net : Type := ...
```

Restaurant Menu

Menu	35€
Entree	Quiche or Salmon
Plat	Pasta or Duck
Dessert	Fruit (Banana or Apple according to season) or Cake (Flan or Chocolate according to Chief's mood)
Sides	Water at will

Restaurant Menu

Menu	35€
Entree	Quiche or Salmon
Plat	Pasta or Duck
Dessert	Fruit (Banana or Apple according to season) or Cake (Flan or Chocolate according to Chief's mood)
Sides	Water at will

$35€ \multimap$

$\multimap$ linear implication, consume its premise (or as $A \implies B = \neg A \vee B$)

Restaurant Menu

Menu	35€
Entree	Quiche or Salmon
Plat	Pasta or Duck
Dessert	Fruit (Banana or Apple according to season) or Cake (Flan or Chocolate according to Chief's mood)
Sides	Water at will

$$35€ \multimap [(Q \,\&\, S)$$

$\multimap$ linear implication, consume its premise (or as $A \implies B = \neg A \vee B$)

$\&$ and where we (the client) choose between two options

Restaurant Menu

Menu	35€
Entree	Quiche or Salmon
Plat	Pasta or Duck
Dessert	Fruit (Banana or Apple according to season) or Cake (Flan or Chocolate according to Chief's mood)
Sides	Water at will

$$35€ \multimap [(Q \,\&\, S) \otimes (P \,\&\, D)$$

$\multimap$ linear implication, consume its premise (or as $A \implies B = \neg A \vee B$)

$\&$ and where we (the client) choose between two options

$\otimes$ and where we get both options

Restaurant Menu

Menu	35€
Entree	Quiche or Salmon
Plat	Pasta or Duck
Dessert	Fruit (Banana or Apple according to season) or Cake (Flan or Chocolate according to Chief's mood)
Sides	Water at will

$$35€ \multimap [(Q \,\&\, S) \otimes (P \,\&\, D) \otimes ((B \oplus A) \,\&\, (F \oplus C))$$

- $\multimap$ linear implication, consume its premise (or as $A \implies B = \neg A \vee B$)
- $\&$ and where we (the client) choose between two options
- $\otimes$ and where we get both options
- $\oplus$ or where we (the client) do not choose between two options

Restaurant Menu

Menu	35€
Entree	Quiche or Salmon
Plat	Pasta or Duck
Dessert	Fruit (Banana or Apple according to season) or Cake (Flan or Chocolate according to Chief's mood)
Sides	Water at will

$$35€ \multimap [(Q \,\&\, S) \otimes (P \,\&\, D) \otimes ((B \oplus A) \,\&\, (F \oplus C)) \otimes !W]$$

- $\multimap$ linear implication, consume its premise (or as $A \implies B = \neg A \vee B$)
- $\&$ and where we (the client) choose between two options
- $\otimes$ and where we get both options
- $\oplus$ or where we (the client) do not choose between two options
- $!$ unlimited resource