Vérification formelle de systèmes par Model-Checking

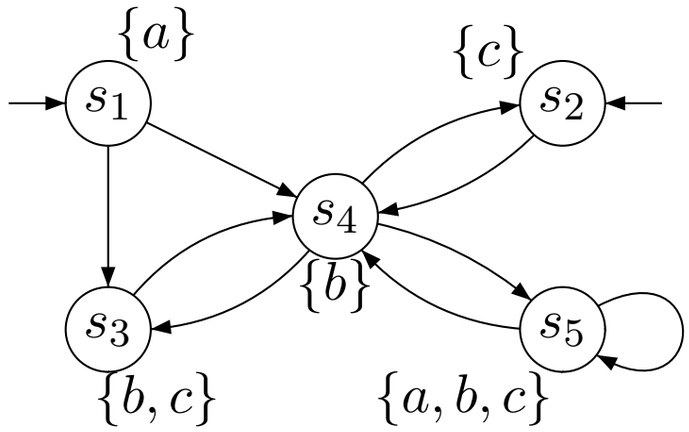
Nathalie Sznajder
Université Pierre et Marie Curie, LIP6

Algorithmes de Model-Checking

Model-Checking de LTL

- Données : Une structure de Kripke $M=(Q, T, A, q_0, AP, l)$ et une formule LTL φ.
- Question : Est-ce que $M \models \varphi$?
  - $M \models \varphi$ ssi $t,0 \models \varphi$ pour toute trace initiale t de M.

Exercice



M⊨φ?

- φ=FGc
- φ=GFc
- φ=Ga
- φ=aU(G(b∨c))
- X¬c→XXc

Model-Checking de LTL: principe

- Soit Σ un alphabet. On note Σ^* l'ensemble des mots finis et Σ^ω les mots infinis.
- Modèles de φ = mots infinis. Soit $[\![\varphi]\!]$ le langage des modèles de la formule : $[\![\varphi]\!]=\{t\in(2^{AP})^\omega \mid t, 0\models \varphi\}$
- Soit $[\![M]\!]$ le langage des traces initiales de M :$[\![M]\!]=\{t\in (2^{AP})^\omega \mid t \text{ est une trace initiale de M}\}$
- Le problème du model-checking revient donc à vérifier si : $[\![M]\!]\subseteq[\![\varphi]\!]$

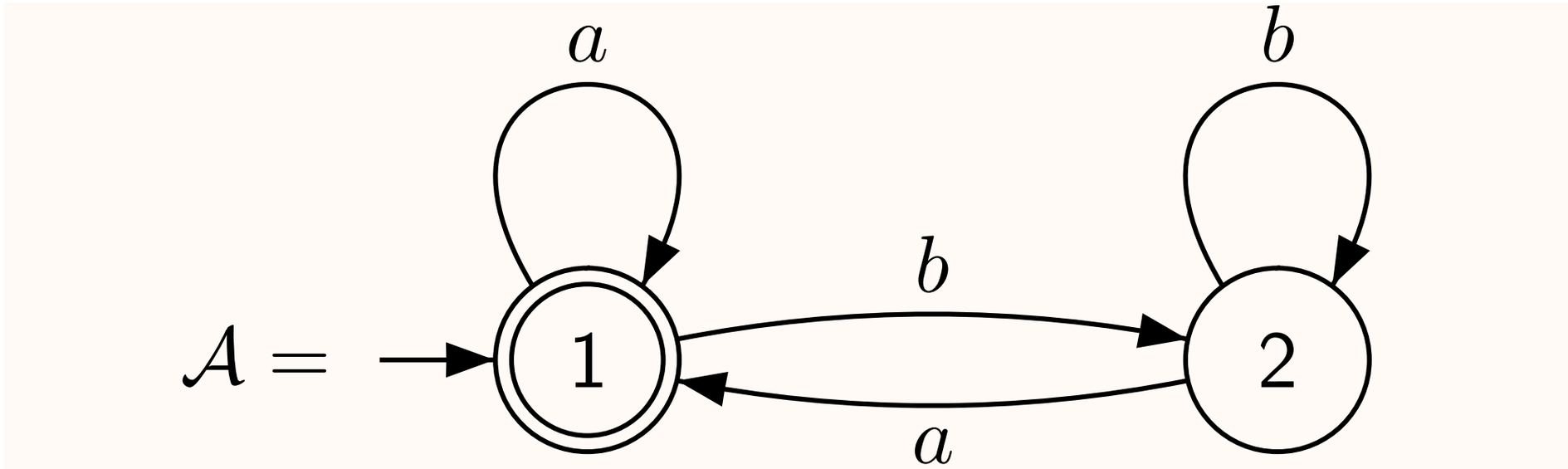
Outil : les automates de Büchi

- Définition : Un automate de Büchi est un n-uplet $A=(Q, \Sigma, I, T, F)$ avec
  - Q un ensemble fini d'états
  - Σ un alphabet fini
  - $I \subseteq Q$ les états initiaux
  - $T \subseteq Q \times \Sigma \times Q$ la relation de transition
  - $F \subseteq Q$ un ensemble d'états acceptants (ou répétés)

Outil : les automates de Büchi

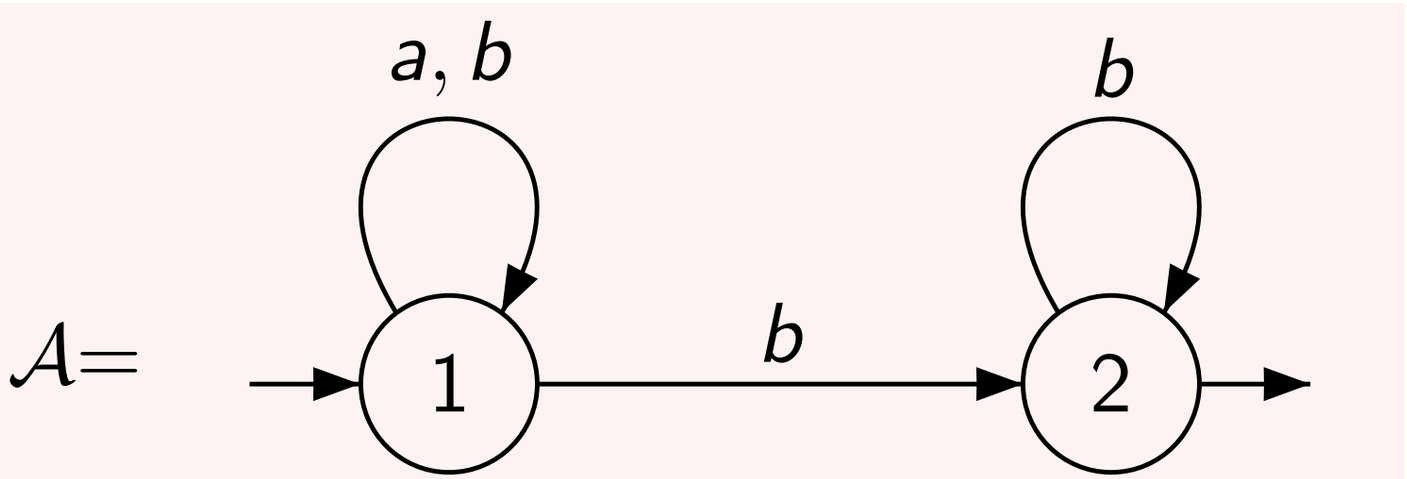
- Une exécution de A sur un mot infini $w=w_0w_1w_2...$ de Σ^ω est une séquence $r=q_0q_1q_2q_3...$ telle que $q_0 \in I$ et $(q_i,w_i,q_{i+1}) \in T$, pour tout $i \geq 0$.
  - r est acceptante si $q_i \in F$ pour un nombre infini de i.
  - w est accepté par A s'il existe une exécution acceptante de A sur w.
  - $L(A)=\{w \in \Sigma^\omega \mid w \text{ accepté par A}\}$.

Automate de Büchi: exemple



$$\mathcal{L}(\mathcal{A}) = \{w \in \{a,b\}^\omega \mid |w|_a = \omega\}$$

Automate de Büchi: exemple



$\mathcal{L}(\mathcal{A})$?

Automates de Büchi non-déterministes

- Les automates de Büchi non déterministes sont plus expressifs que les automates de Büchi déterministes
- Les langages reconnus par un NBA forment les ω-réguliers
- Toute formule de LTL peut être reconnue par un NBA

Les automates de Büchi pour LTL

- Définition : Un automate de Büchi est un n-uplet $A=(Q, \Sigma, I, T, F)$ avec
  - Q un ensemble fini d'états
  - Σ un alphabet fini
  - $I \subseteq Q$ les états initiaux
  - $T \subseteq Q \times \Sigma \times Q$ la relation de transition
  - $F \subseteq Q$ un ensemble d'états acceptants (ou répétés)

Les automates de Büchi pour LTL

- Définition : Un automate de Büchi est un n-uplet $A=(Q, \Sigma, I, T, F)$ avec
  - Q un ensemble fini d'états
  - Σ un alphabet fini $\Sigma = 2^{AP}$
  - $I \subseteq Q$ les états initiaux
  - $T \subseteq Q \times \Sigma \times Q$ la relation de transition
  - $F \subseteq Q$ un ensemble d'états acceptants (ou répétés)

Exercice

- Exemple : automate de Büchi reconnaissant p, Xp.
- Construire des automates de Büchi reconnaissant Fp, XXp, Gp, FGp, GFp, pUq, pRq.

Automates de Büchi et LTL

- Les formules de LTL sont moins expressives que les automates de Büchi
- Exemple : «Un instant sur deux, l'événement a arrive.» est une propriété ω régulière non exprimable en LTL

Automates de Büchi

Théorème : Les automates de Büchi sont clos par union, intersection, et complément.

Théorème : on peut tester le vide d'un automate de Büchi.

Automates de Büchi - Test du vide

- Chercher si un état acceptant est accessible depuis l'état initial
- Chercher si cet état appartient à un cycle

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ (assez facile)
  - Transformer φ en un automate A_φ tel que $L(A_\varphi)=[\![\varphi]\!]$ (plus difficile)
  - Tester si $L(A_M) \subseteq L(A_\varphi)$, i.e., si $L(A_M) \cap L(A_\varphi)^c = \varnothing$.

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un [illegible] tel que $L(A_M)=[\![M]\!]$ (ass[illegible]
  - Transf[illegible] automate A_φ tel que L([illegible] difficile)
  - Tes[illegible] si $L(A_M) \subseteq L(A_\varphi)$, i.e., si $L(A_M) \cap L(A_\varphi)^c = \varnothing$.

Difficile de complémenter un automate de Büchi!!

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Transformer φ en un automate de Büchi

I. Automates de Büchi généralisés

II. Réduire la formule

    I. Forme normale négative

    II. Réduire les connecteurs temporels

III. Construire un graphe

IV. Transformation en automate de Büchi

Transformer φ en un automate de Büchi

I. Automates de Büchi généralisés

II. Réduire la formule

  I. Forme normale négative

  II. Réduire les connecteurs temporels

III. Transformation en automate de Büchi généralisé

Automates de Büchi généralisés

- Définition : Un automate de Büchi généralisé est un n-uplet A=(Q, Σ, I, T, F) avec
  - Q un ensemble fini d'états
  - Σ un alphabet fini
  - I⊆Q les états initiaux
  - T⊆Q x Σ x Q la relation de transition
  - $F=\{F_1,F_2,...,F_k\}\subseteq 2^Q$ un ensemble d'ensemble d'états acceptants (ou répétés)

Automates de Büchi généralisés

- Une exécution de A sur un mot infini $w=w_0w_1w_2...$ de Σ^ω est une séquence $r=q_0q_1q_2q_3...$ telle que $q_0 \in I$ et $(q_i,w_i,q_{i+1}) \in T$, pour tout $i \geq 0$.

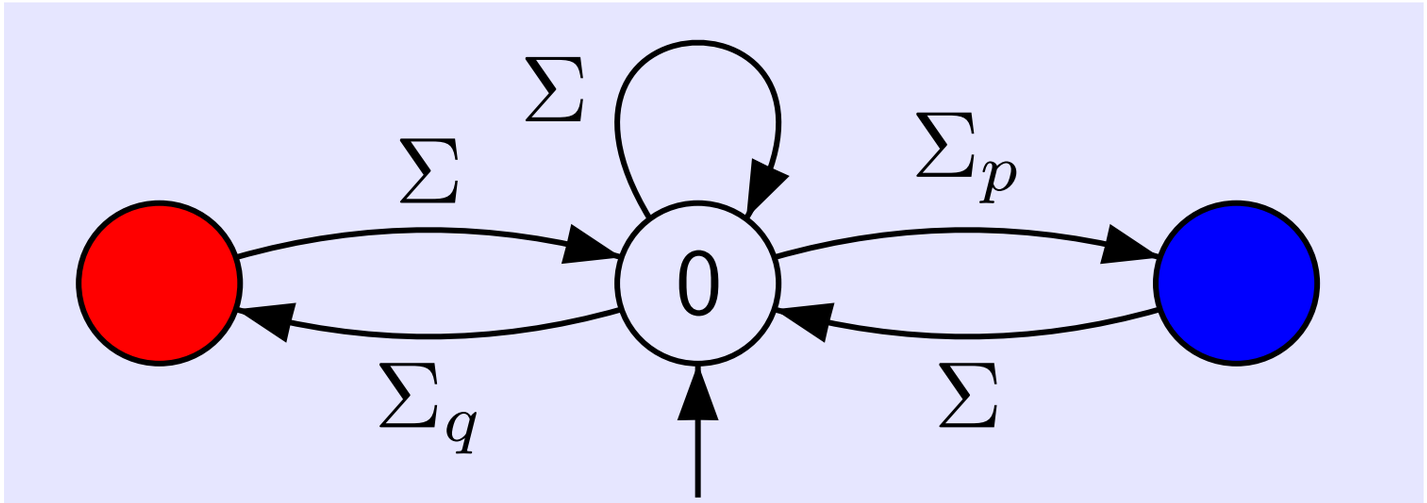
- r est acceptante si pour tout $\mathcal{F} \in F$, $q_i \in \mathcal{F}$ pour un nombre infini de i.

- w est accepté par A s'il existe une exécution acceptante de A sur w.

- $L(A)=\{w \in \Sigma^\omega \mid w \text{ accepté par A}\}$.

Automates de Büchi généralisés : exemple

GFp∧GFq:



Des ABG aux AB

Théorème : Tout automate de Büchi généralisé A peut être transformé en un automate de Büchi A' tel que L(A)=L(A')

Transformer φ en un automate de Büchi

I. Automates de Büchi généralisés

II. Réduire la formule

  I. Forme normale négative

  II. Réduire les connecteurs temporels

III. Transformation en automate de Büchi généralisé

Forme normale négative

$\varphi ::= \bot \mid \top \mid p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid$

$X\varphi \mid \varphi U \varphi \mid \varphi R \varphi$

- $\neg\neg p = p$
- $\neg(\varphi_1 \vee \varphi_2) = \neg\varphi_1 \wedge \neg\varphi_2$
- $\neg(\varphi_1 \wedge \varphi_2) = \neg\varphi_1 \vee \neg\varphi_2$
- $\neg(X\varphi) = X(\neg\varphi)$
- $\neg(\varphi_1 U \varphi_2) = \neg\varphi_1 R \neg\varphi_2$
- $\neg(\varphi_1 R \varphi_2) = \neg\varphi_1 U \neg\varphi_2$

Exercice

- Transformer G(p→Fq) en forme normale négative

Réduire les connecteurs temporels

- Idée : Un état de notre graphe va représenter l'ensemble des propositions atomiques vérifiées au prochain instant de la séquence, et l'ensemble des sous-formules qu'il «promet» de vérifier à l'état suivant.
- Pour cela, on ne veut que des propositions atomiques (ou négations), et des sous-formules commençant par X (next).

Réduire les connecteurs temporels

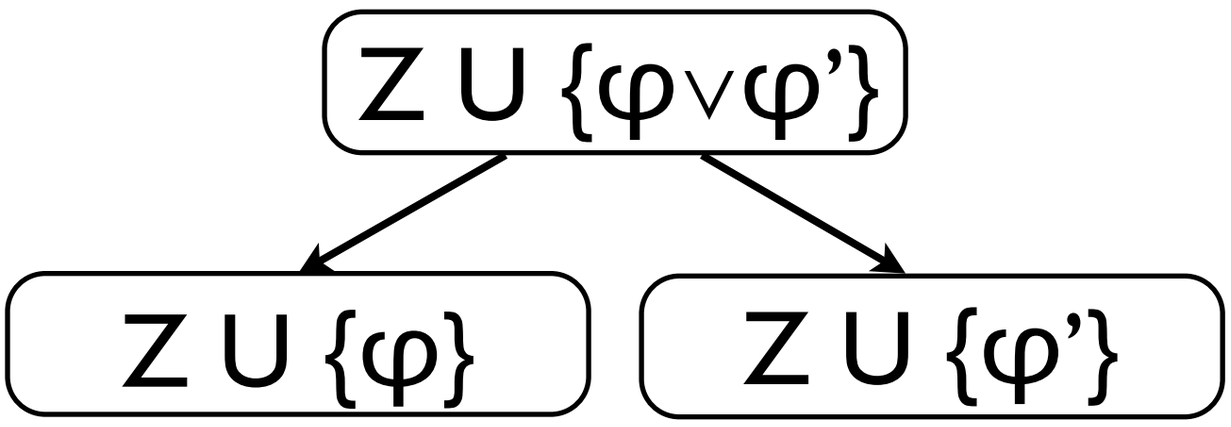
- Un ensemble Z de formules en forme normale négative est réduit si
  1. pour tout $z \in Z$, z est de la forme p, $\neg p$ ou X(z')
  2. il est cohérent : $\bot \notin Z$, $\{p, \neg p\} \not\subseteq Z$, pour tout $p \in AP$.

Réduire les connecteurs temporels

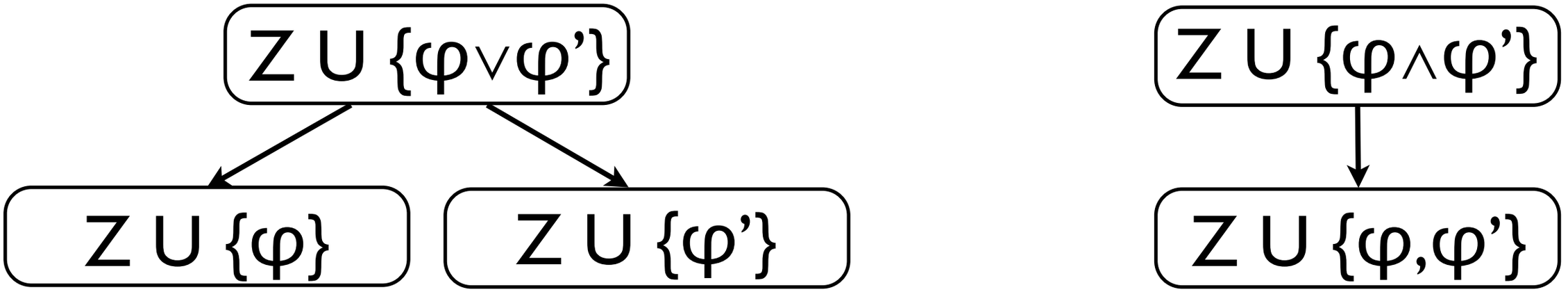
- On utilise les équivalences suivantes :
  - $\varphi U \varphi' \equiv \varphi' \vee (\varphi \wedge X(\varphi U \varphi'))$
  - $\varphi R \varphi' \equiv (\varphi \wedge \varphi') \vee (\varphi' \wedge X(\varphi R \varphi'))$

Construction du graphe de réduction

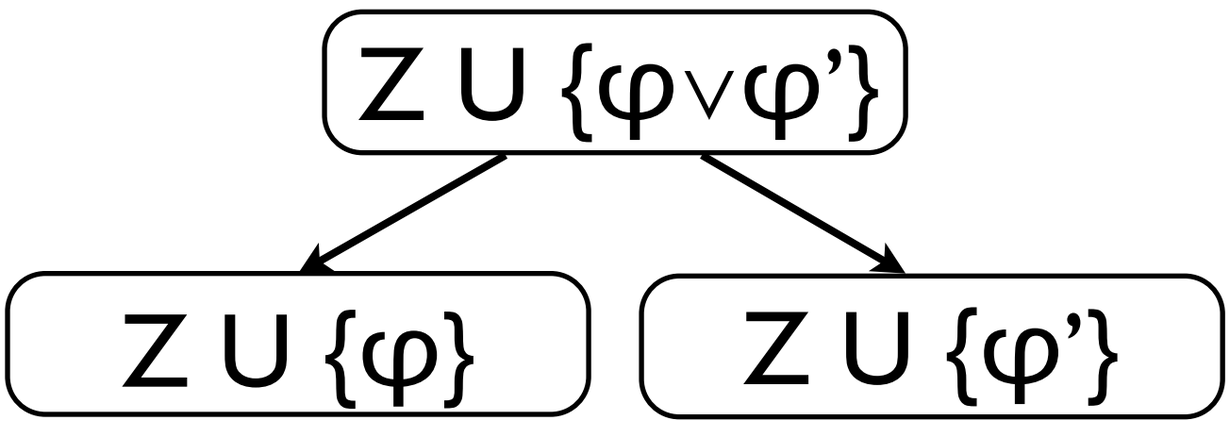
Construction du graphe de réduction

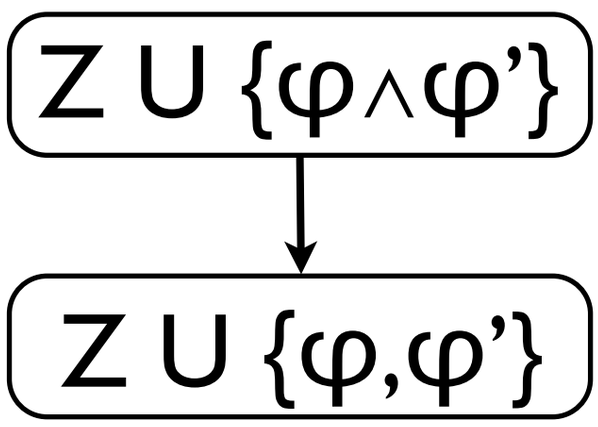


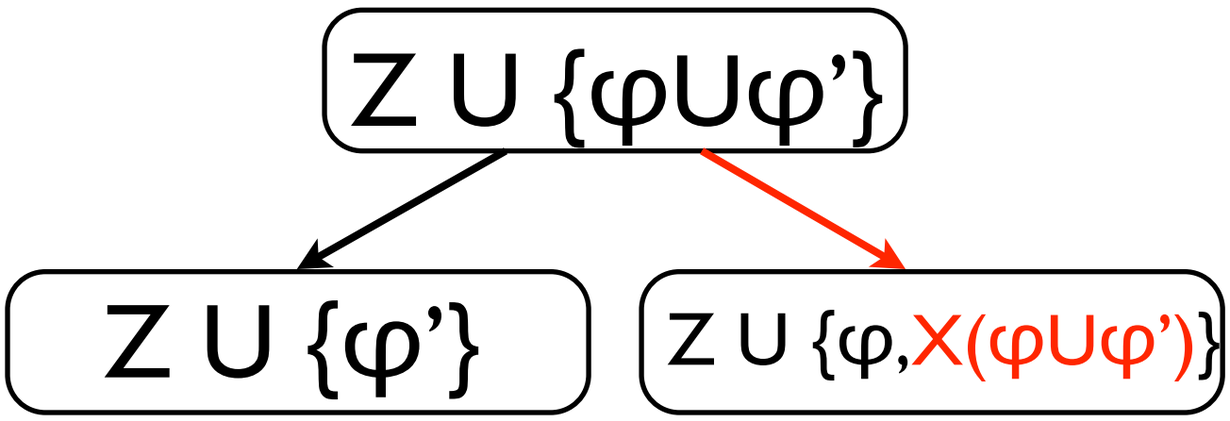
Construction du graphe de réduction



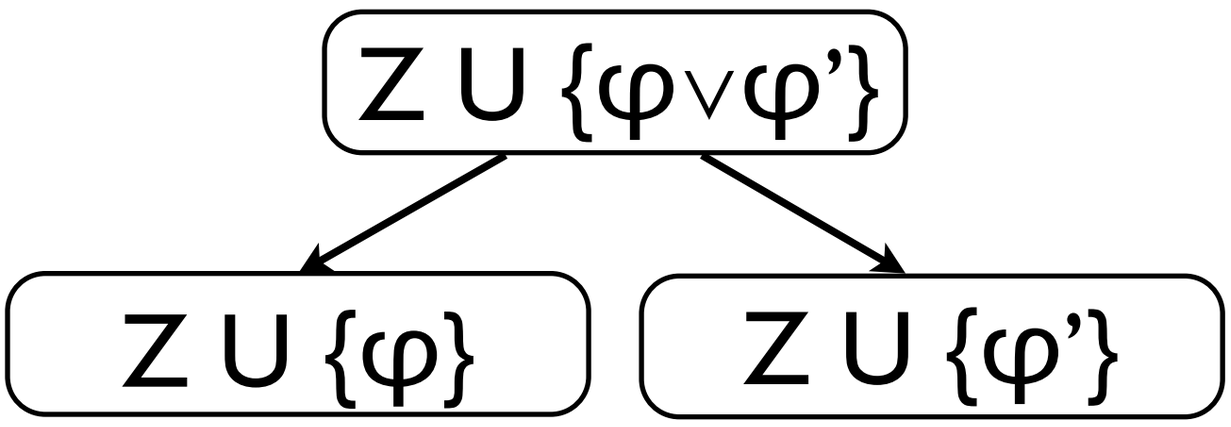
Construction du graphe de réduction

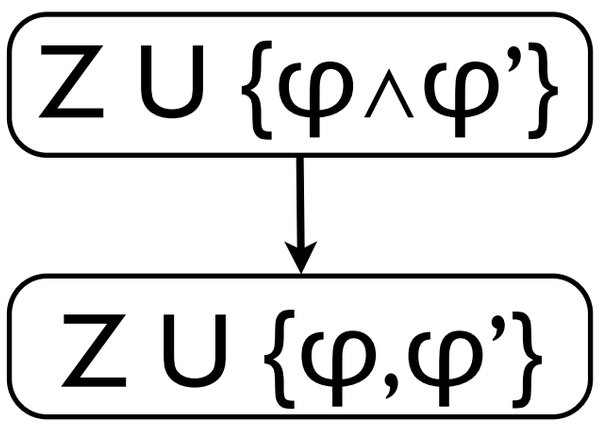


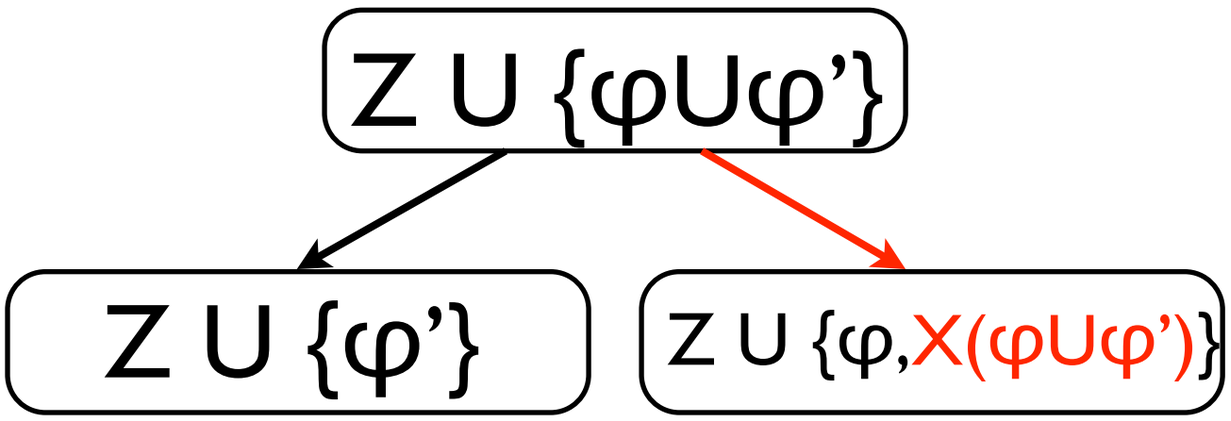


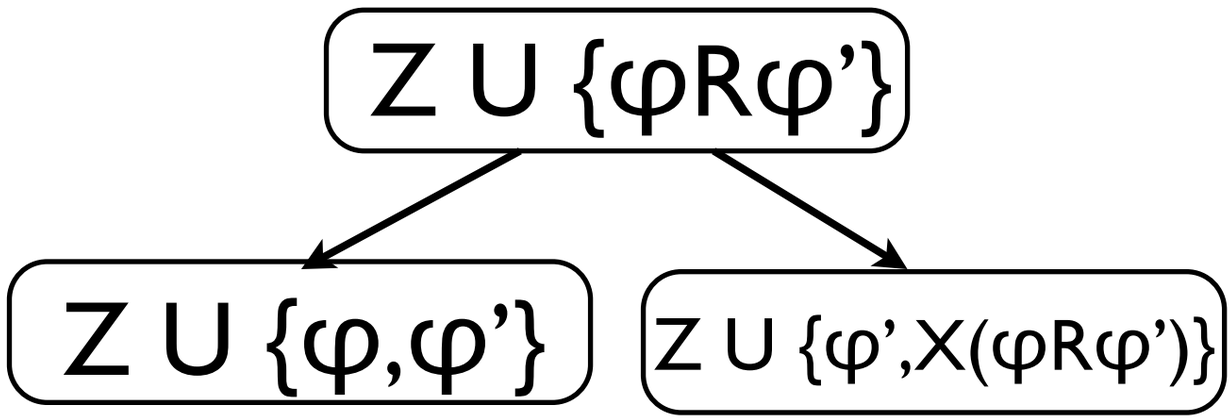


Construction du graphe de réduction









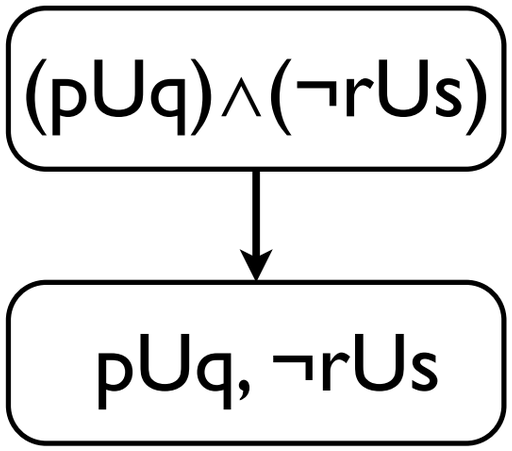
Exemple de réduction d'une formule

$(pUq)\wedge(\neg rUs)$

$(pUq)\wedge(\neg rUs)$

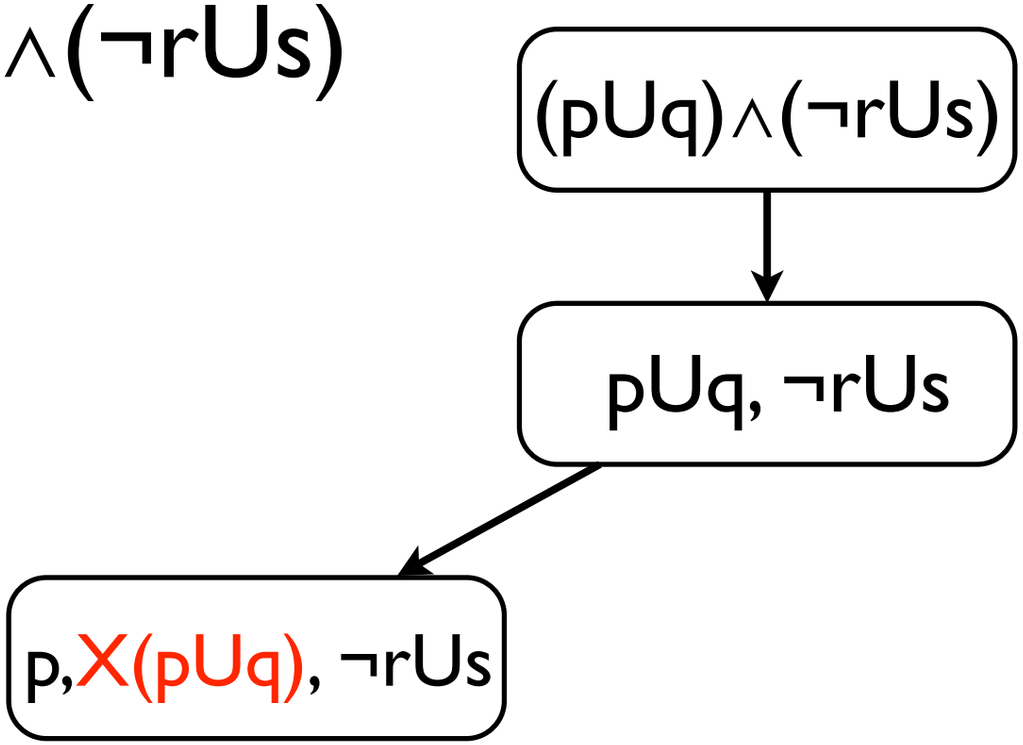
Exemple de réduction d'une formule

$(pUq) \wedge (\neg rUs)$



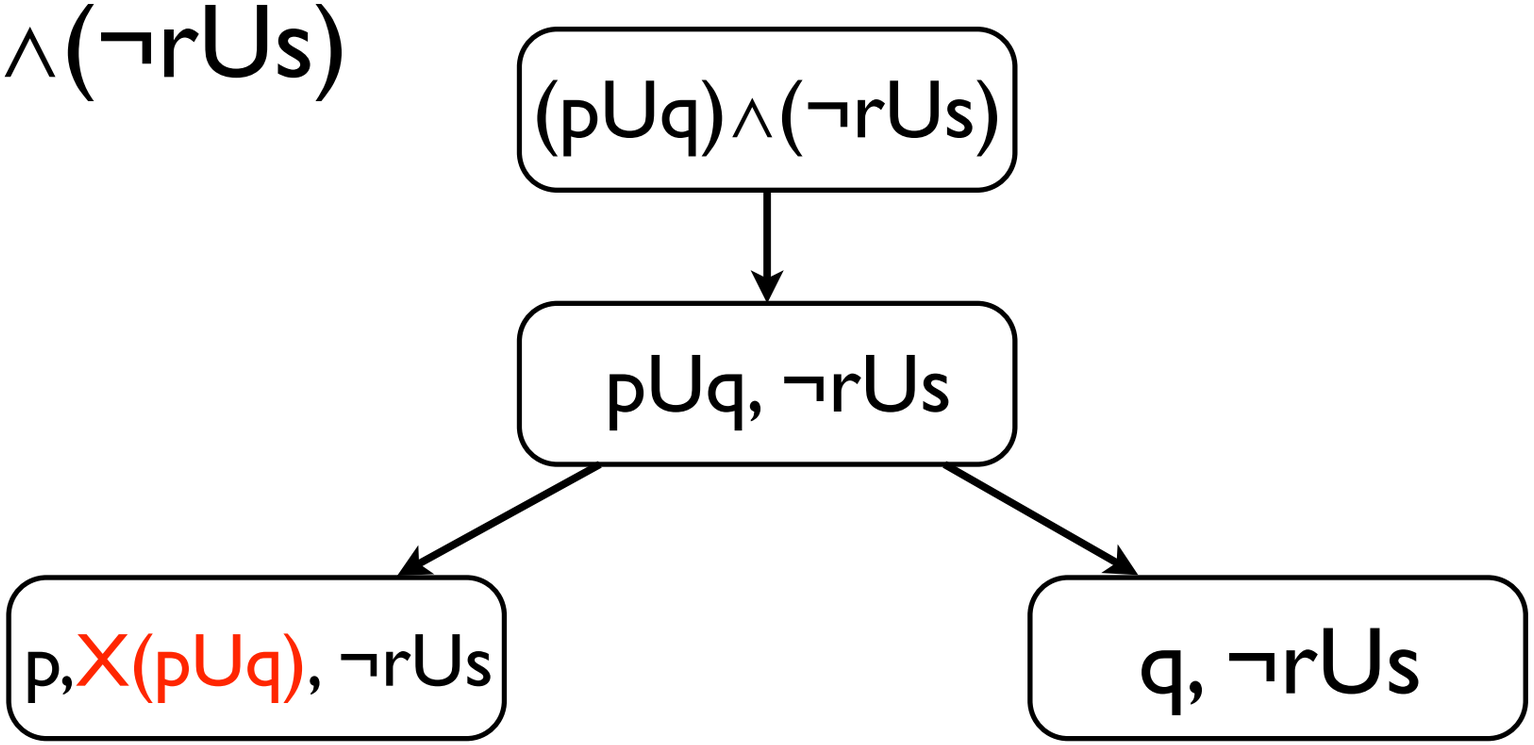
Exemple de réduction d'une formule

$(pUq) \wedge (\neg rUs)$



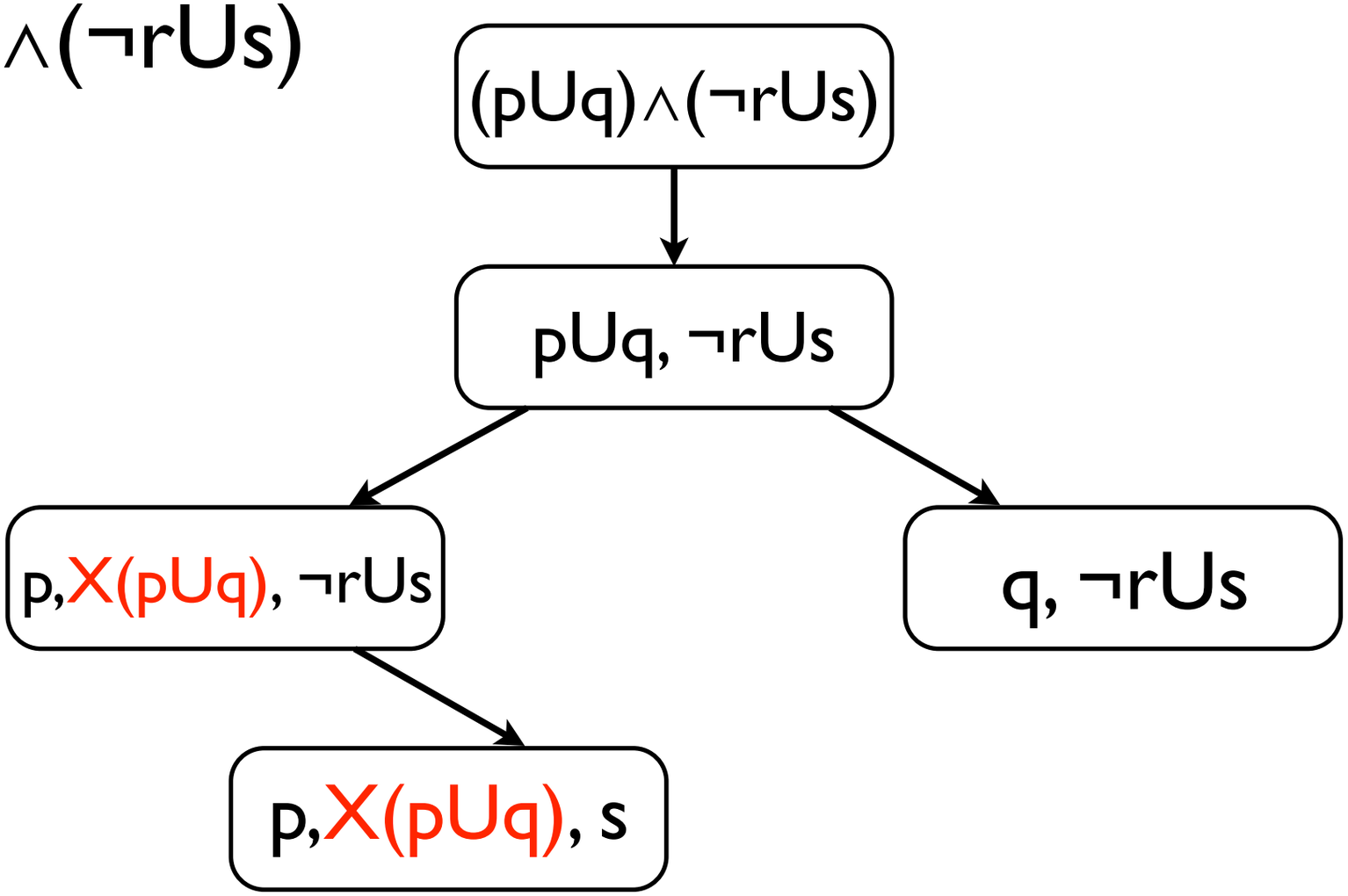
Exemple de réduction d'une formule

$(pUq) \wedge (\neg rUs)$



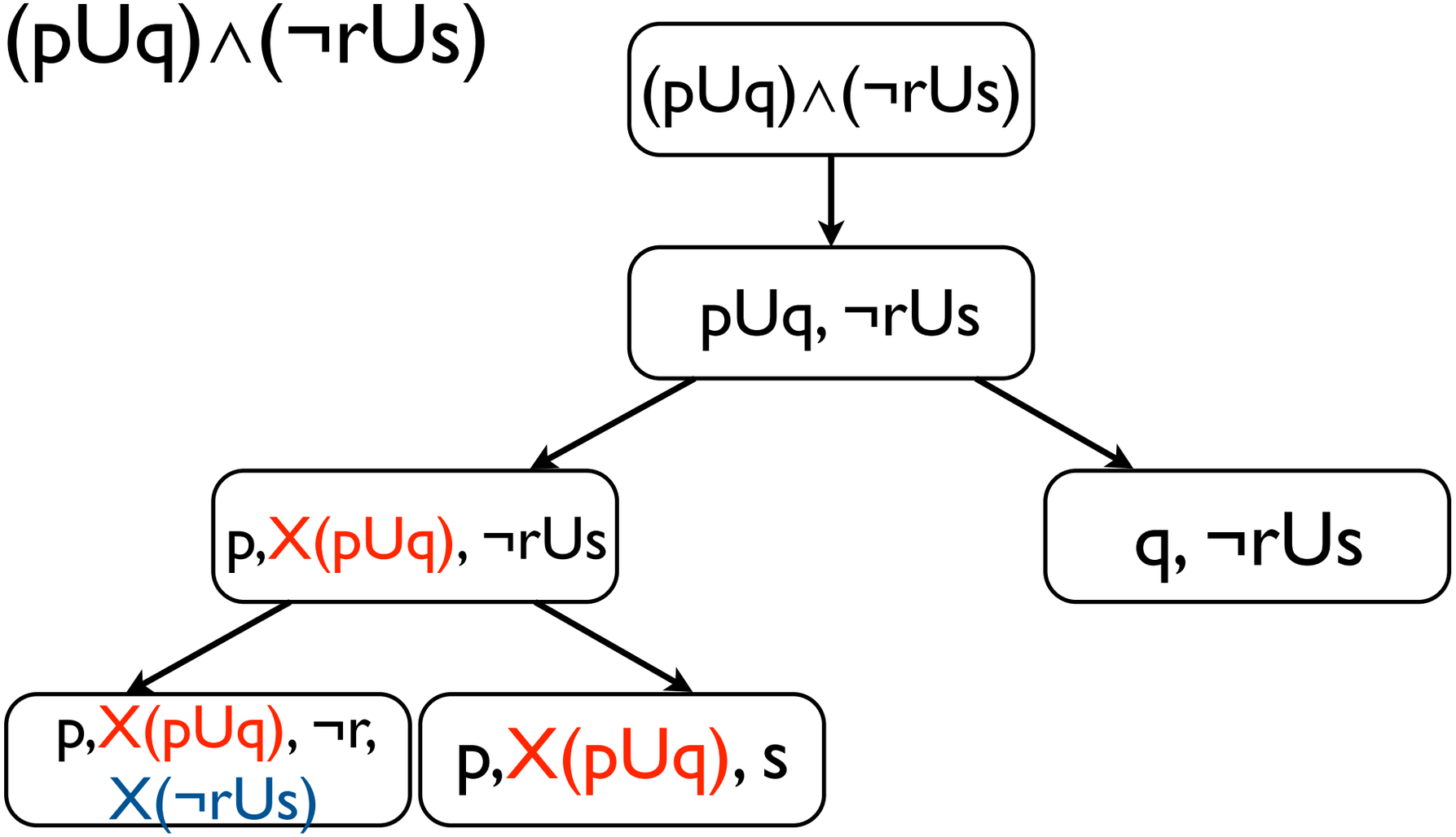
Exemple de réduction d'une formule

$(pUq) \wedge (\neg rUs)$



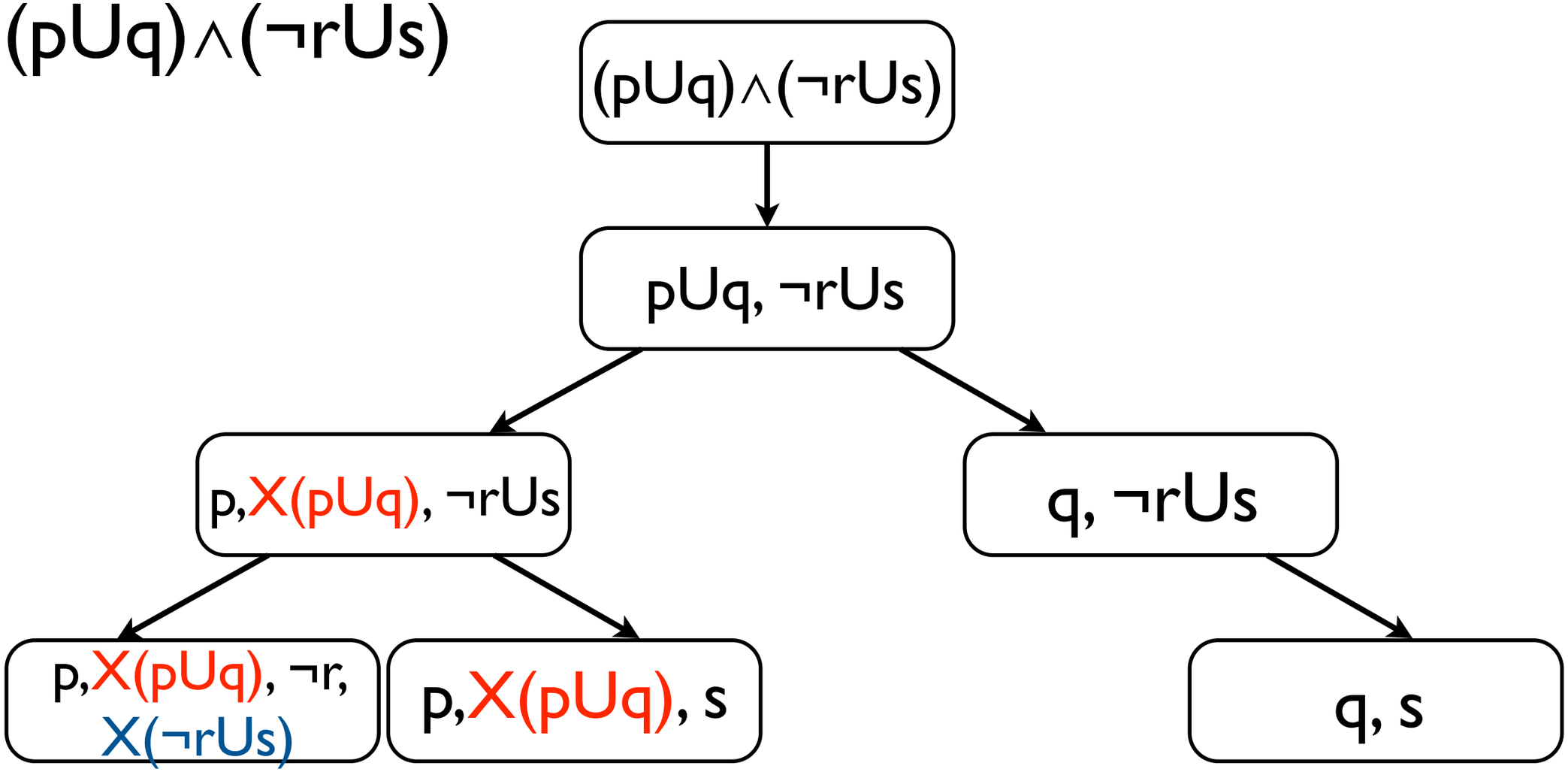
Exemple de réduction d'une formule

$(pUq) \wedge (\neg rUs)$



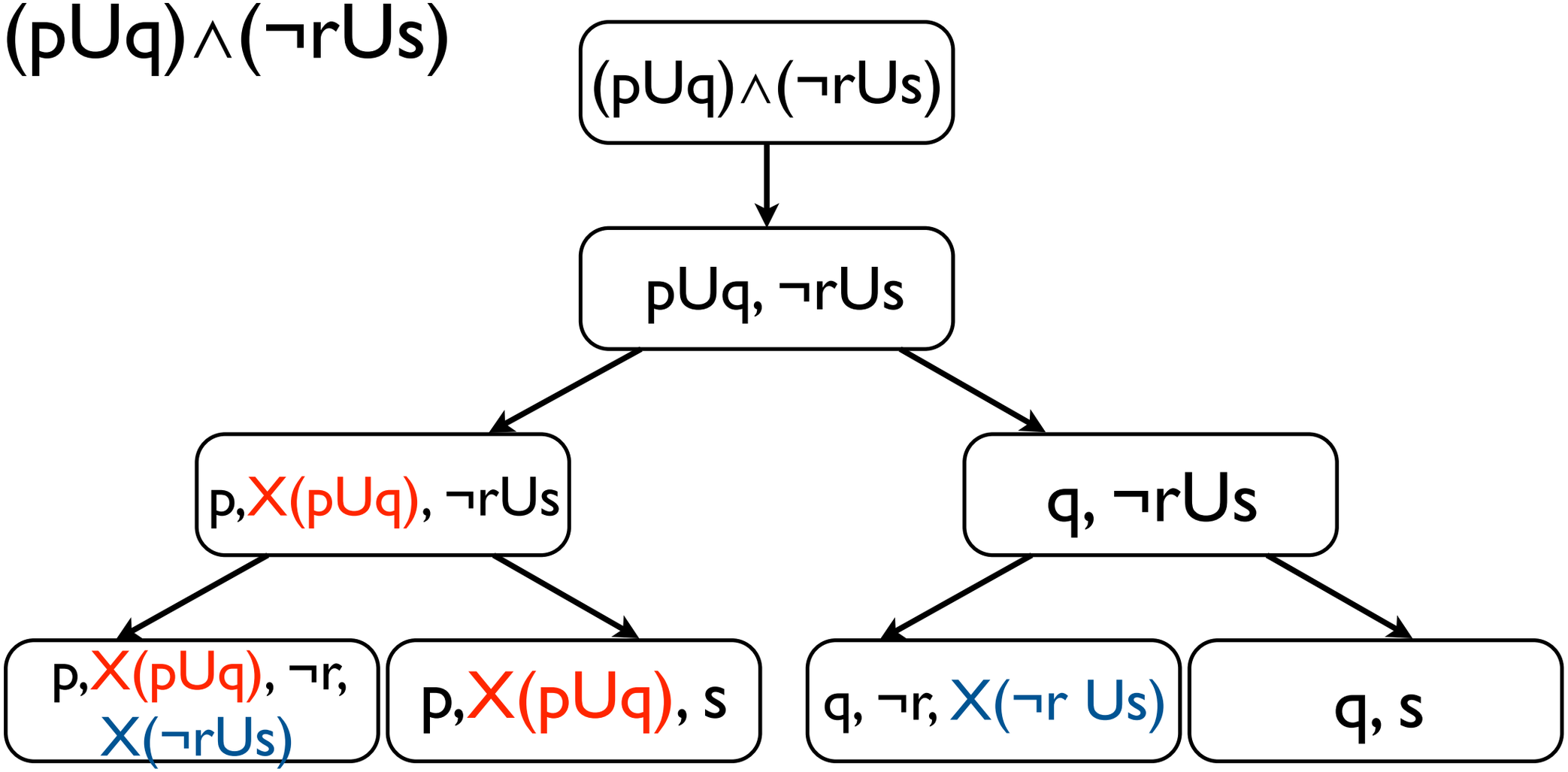
Exemple de réduction d'une formule

$(pUq) \wedge (\neg rUs)$



Exemple de réduction d'une formule

$(pUq) \wedge (\neg rUs)$



Transformer φ en un automate de Büchi

I. Automates de Büchi généralisés

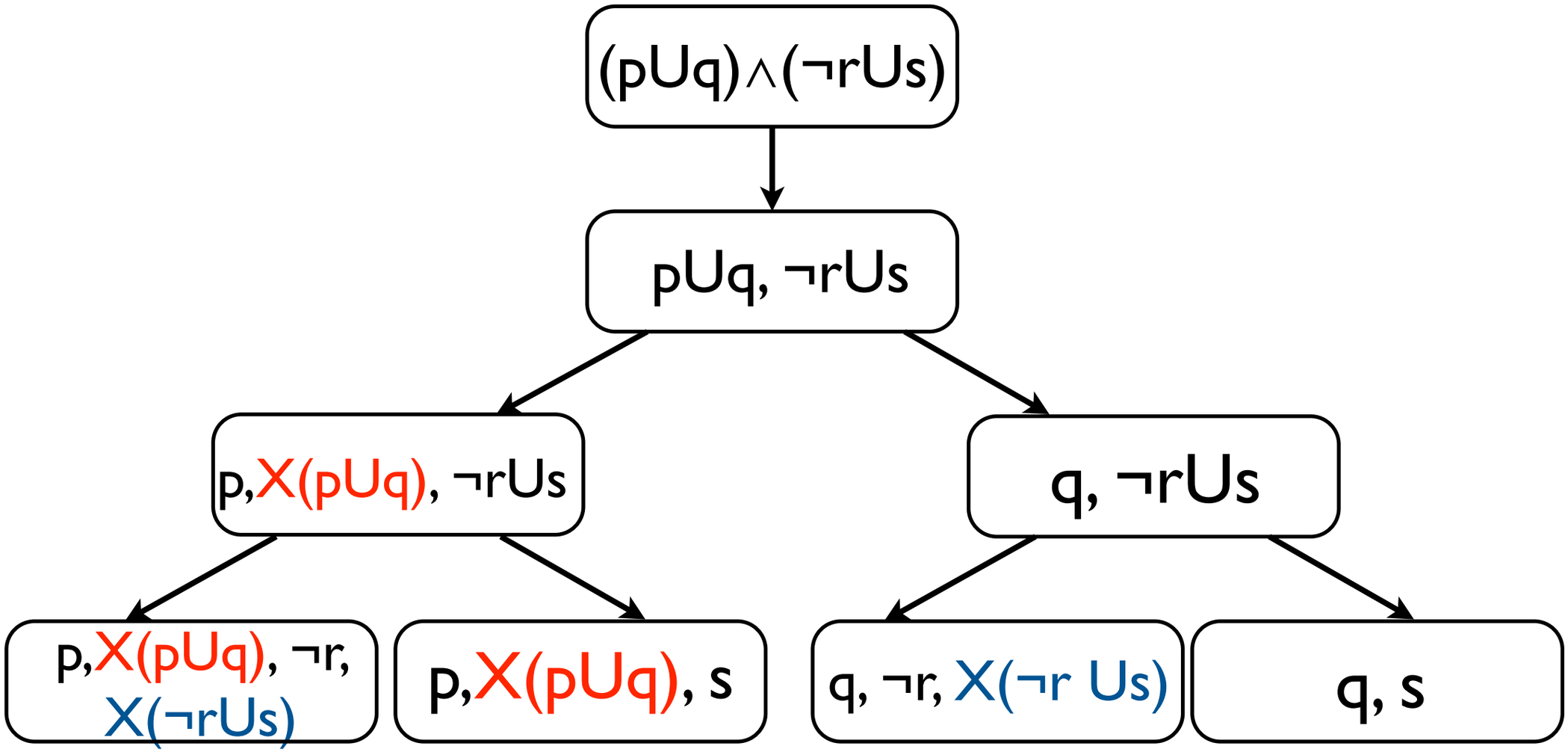
II. Réduire la formule

  I. Forme normale négative

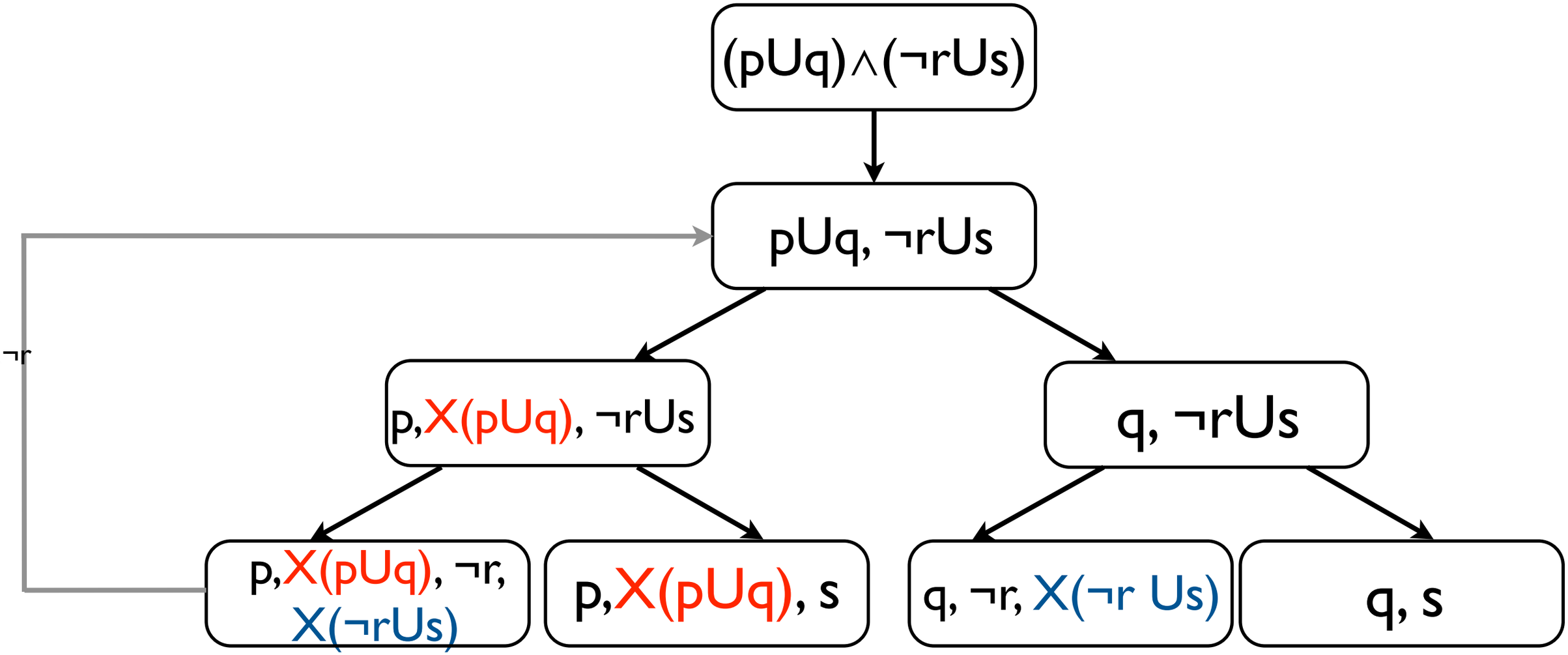
  II. Réduire les connecteurs temporels

III. Transformation en automate de Büchi généralisé

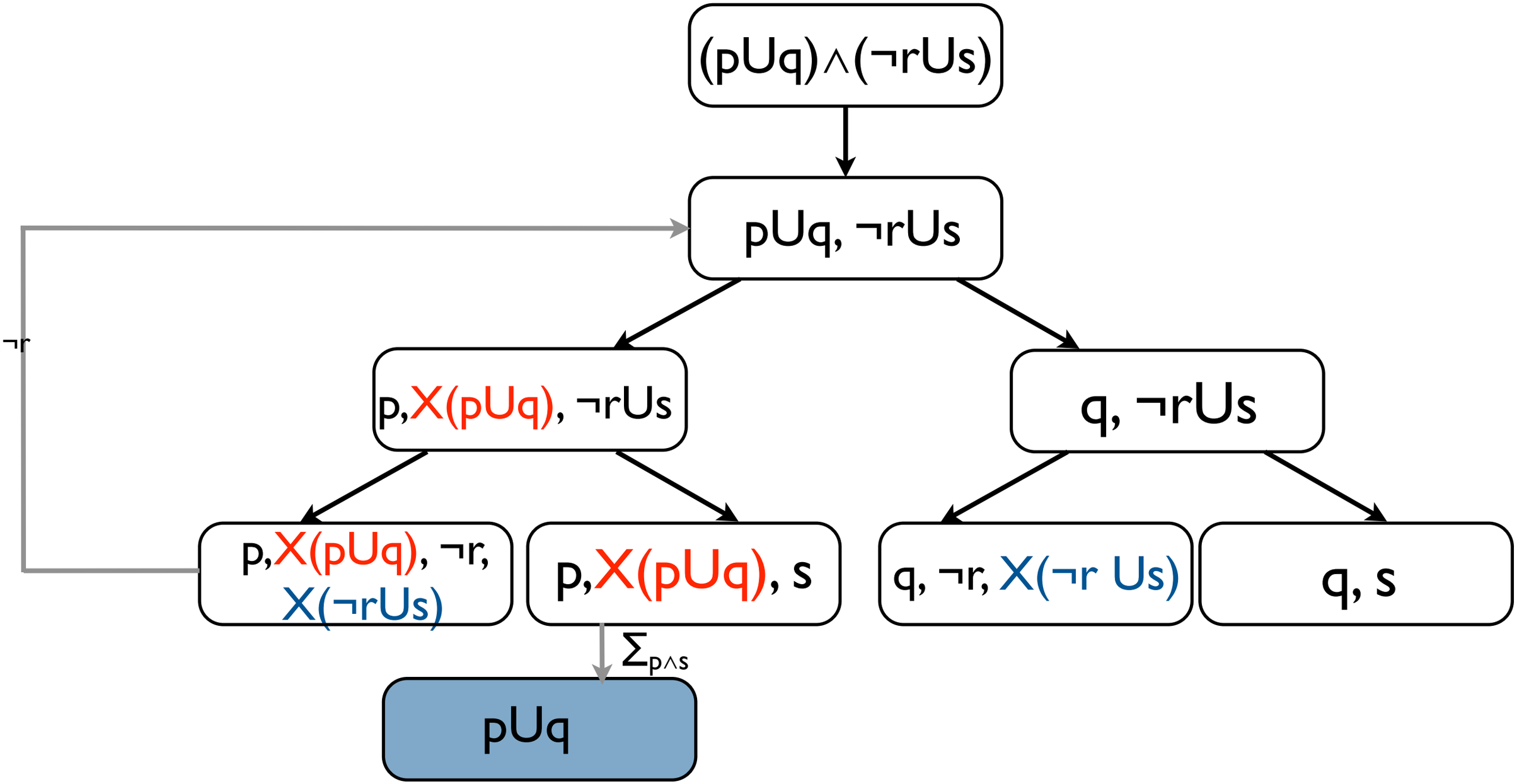
Exemple (suite)



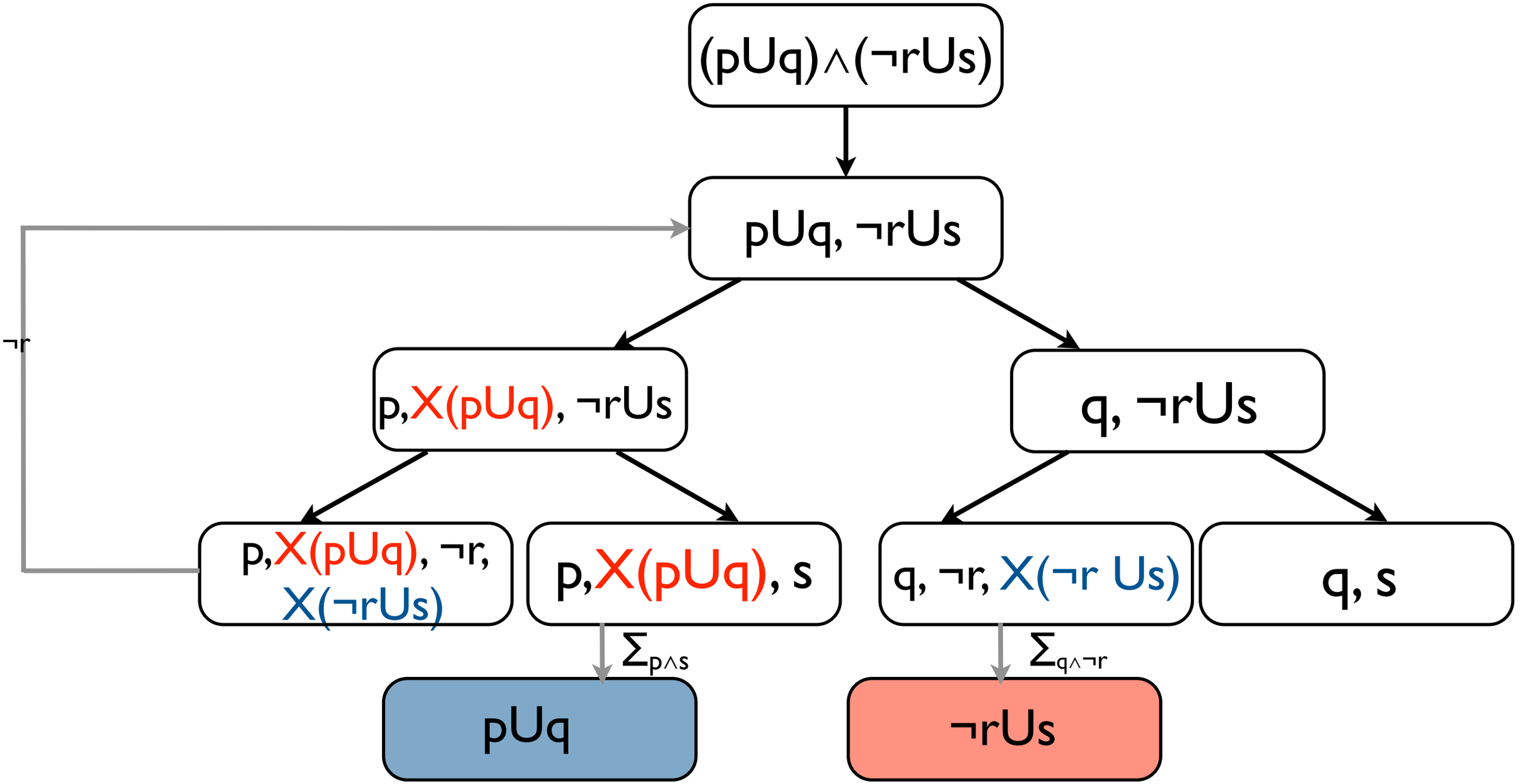
Exemple (suite)



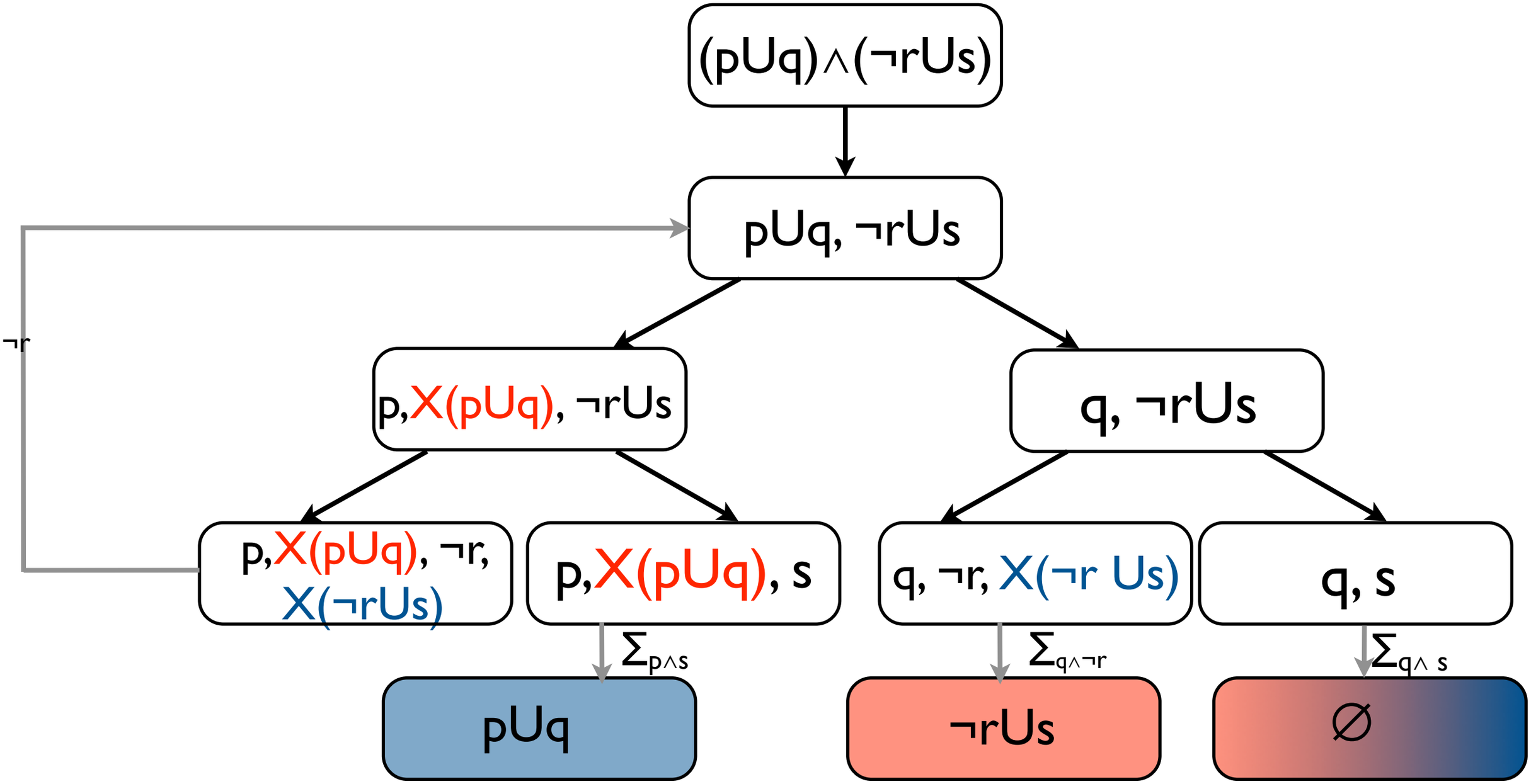
Exemple (suite)



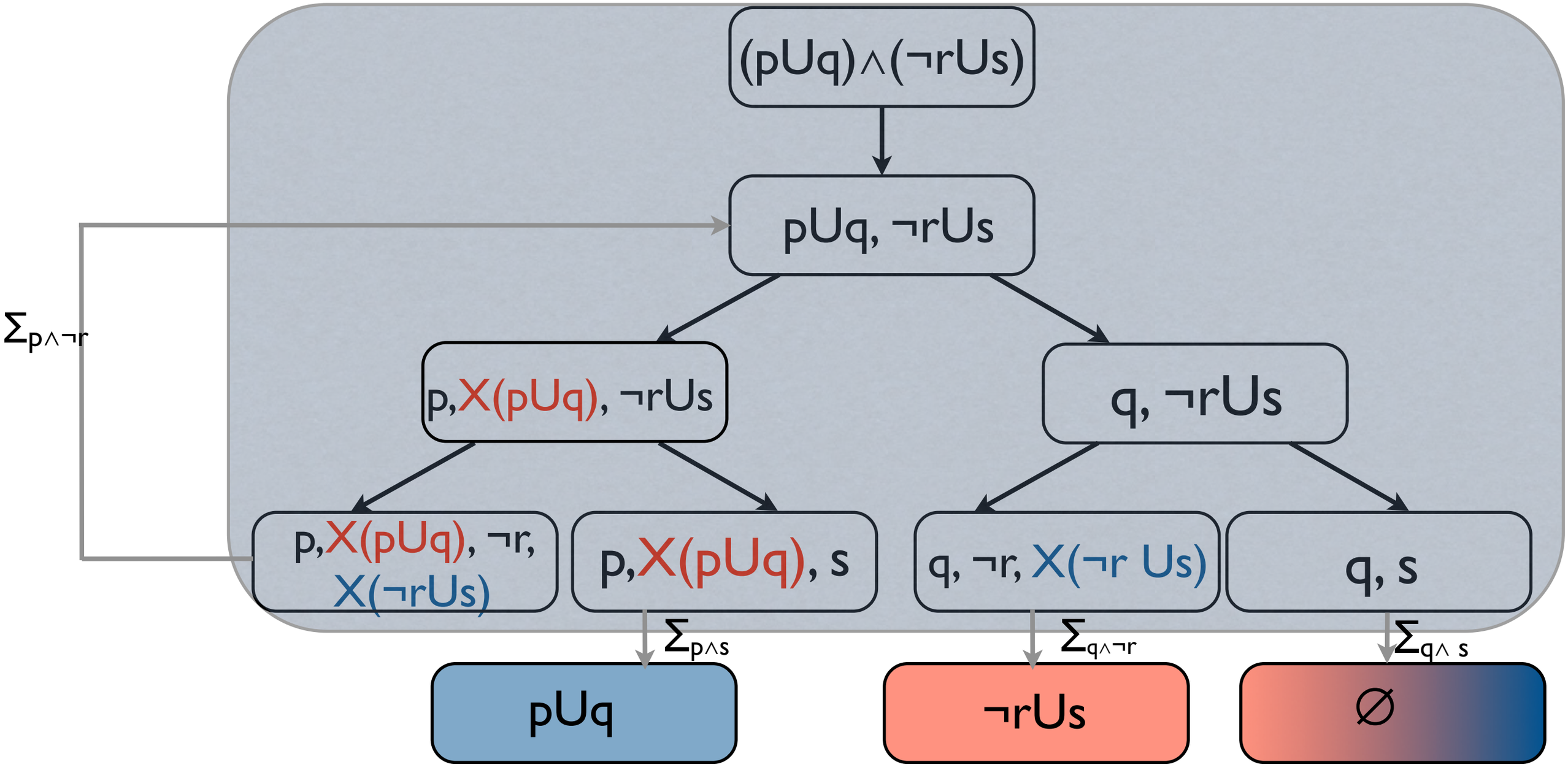
Exemple (suite)



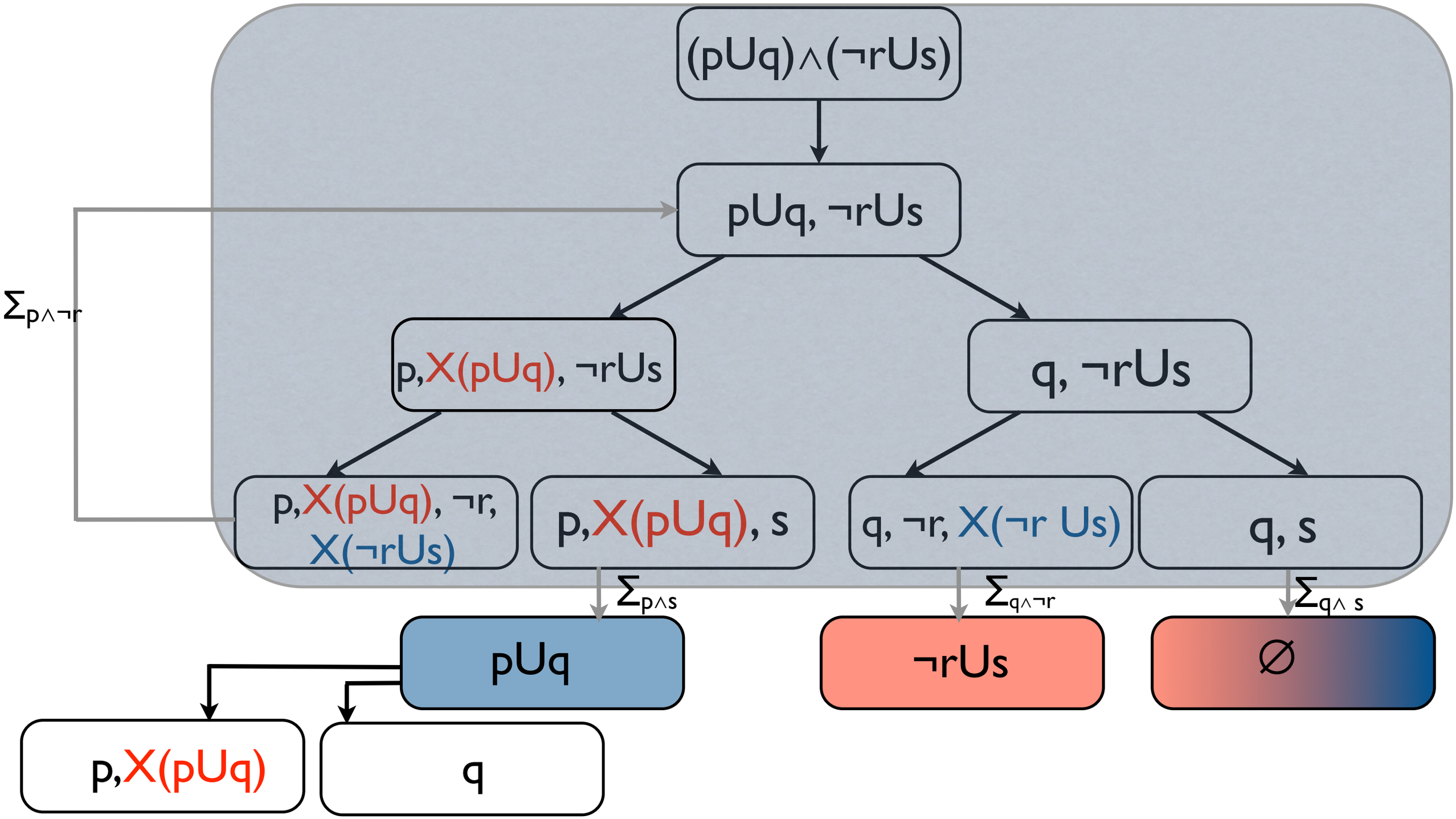
Exemple (suite)



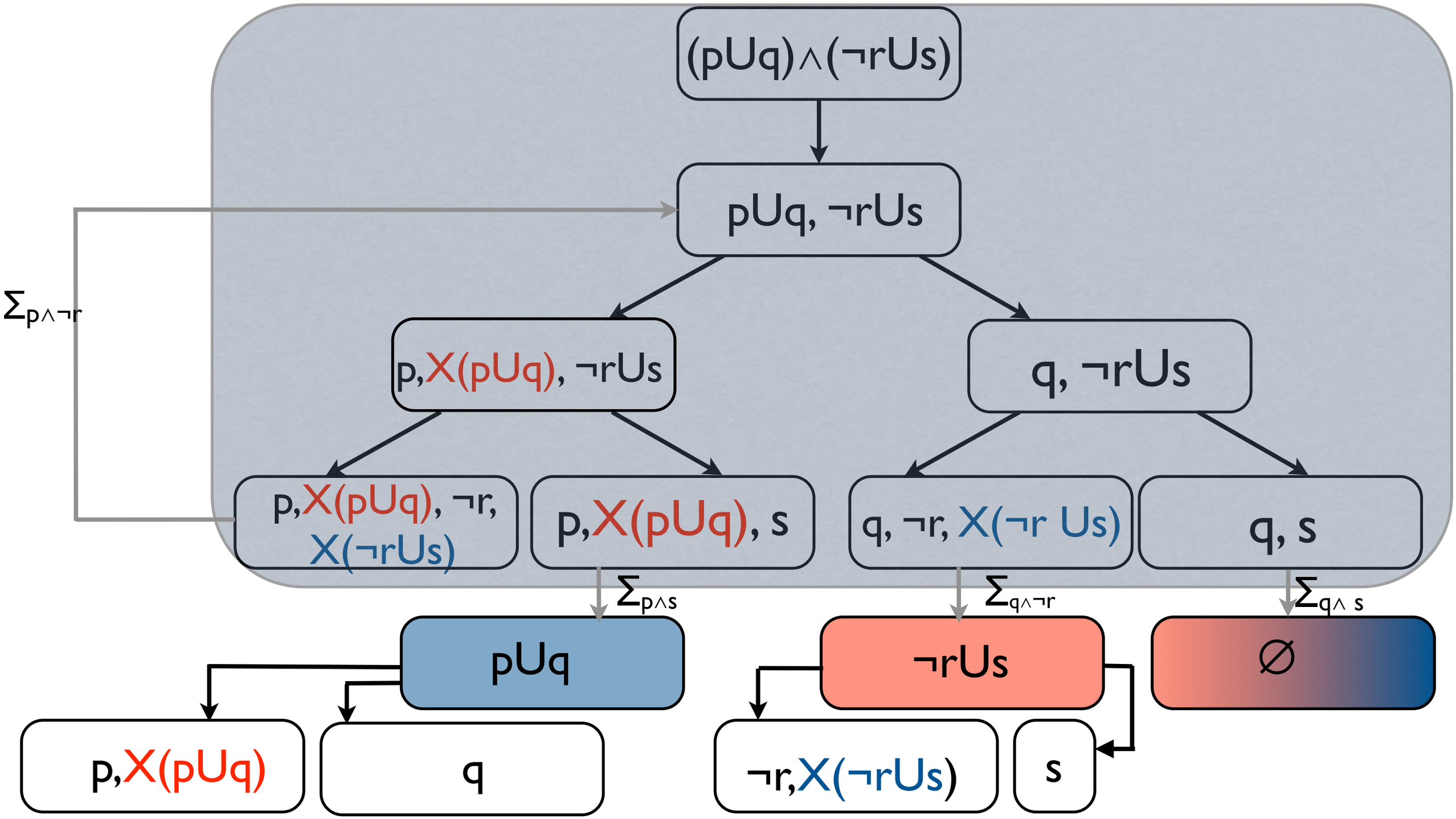
Exemple (suite)



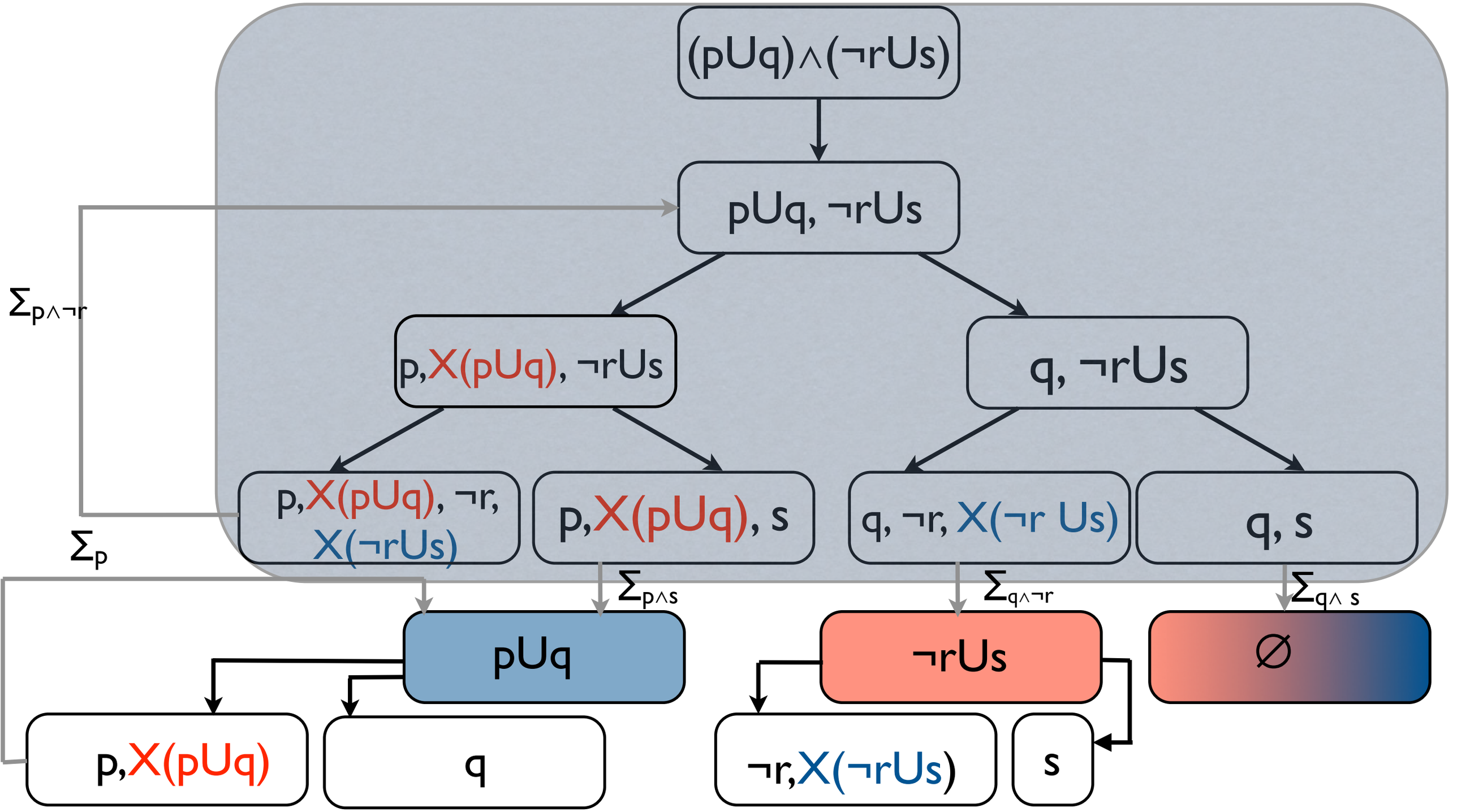
Exemple (suite)



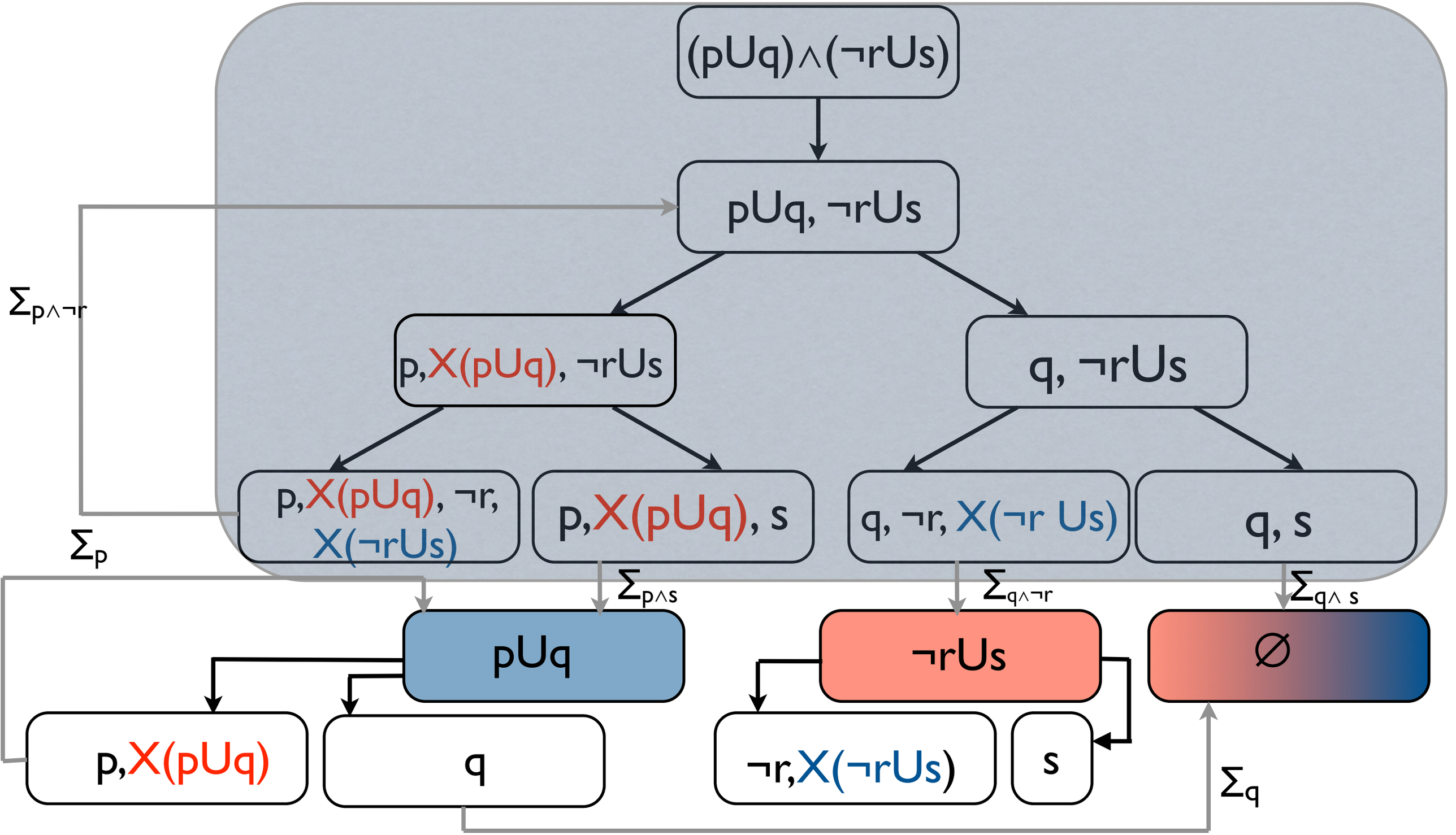
Exemple (suite)



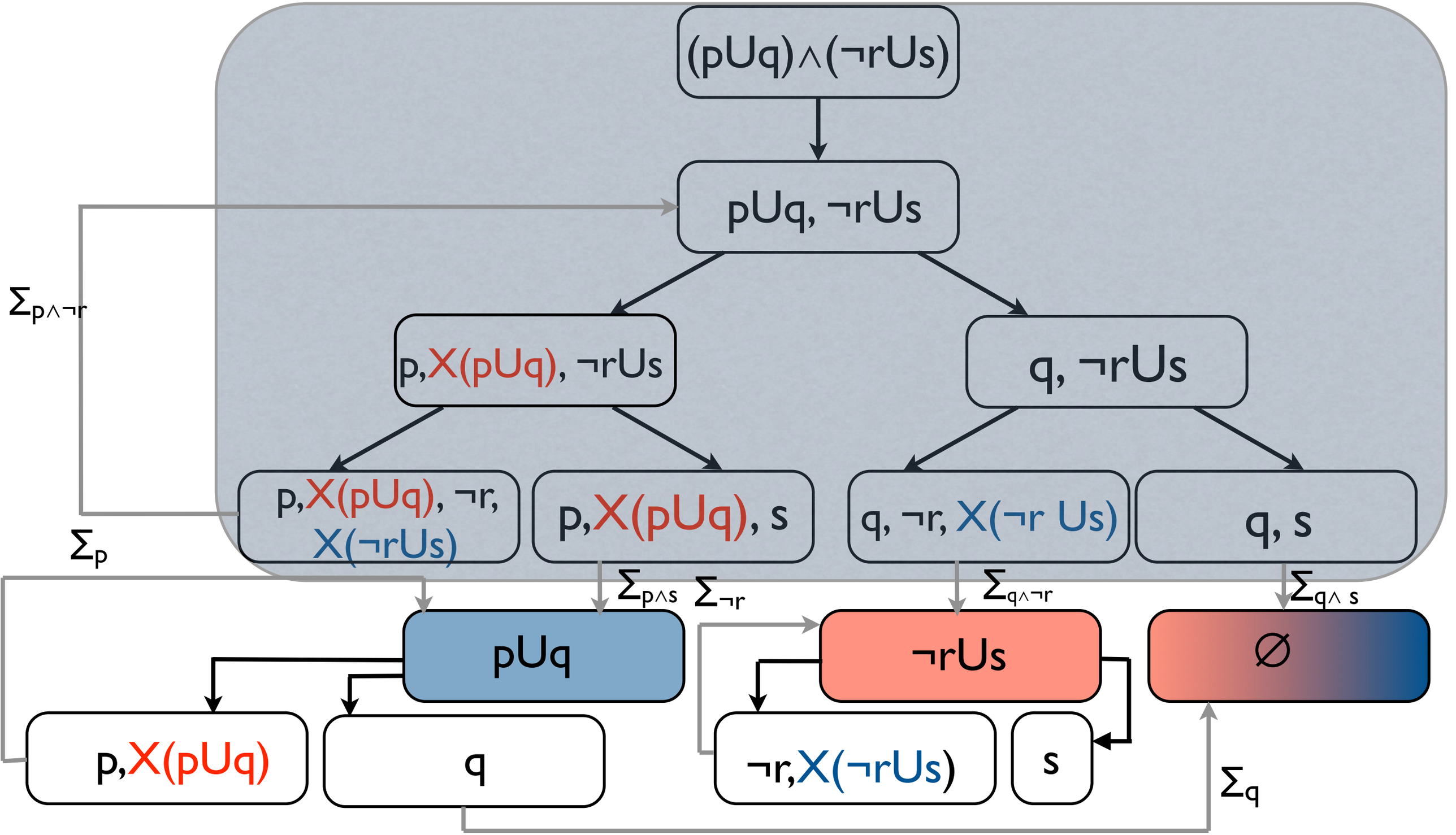
Exemple (suite)



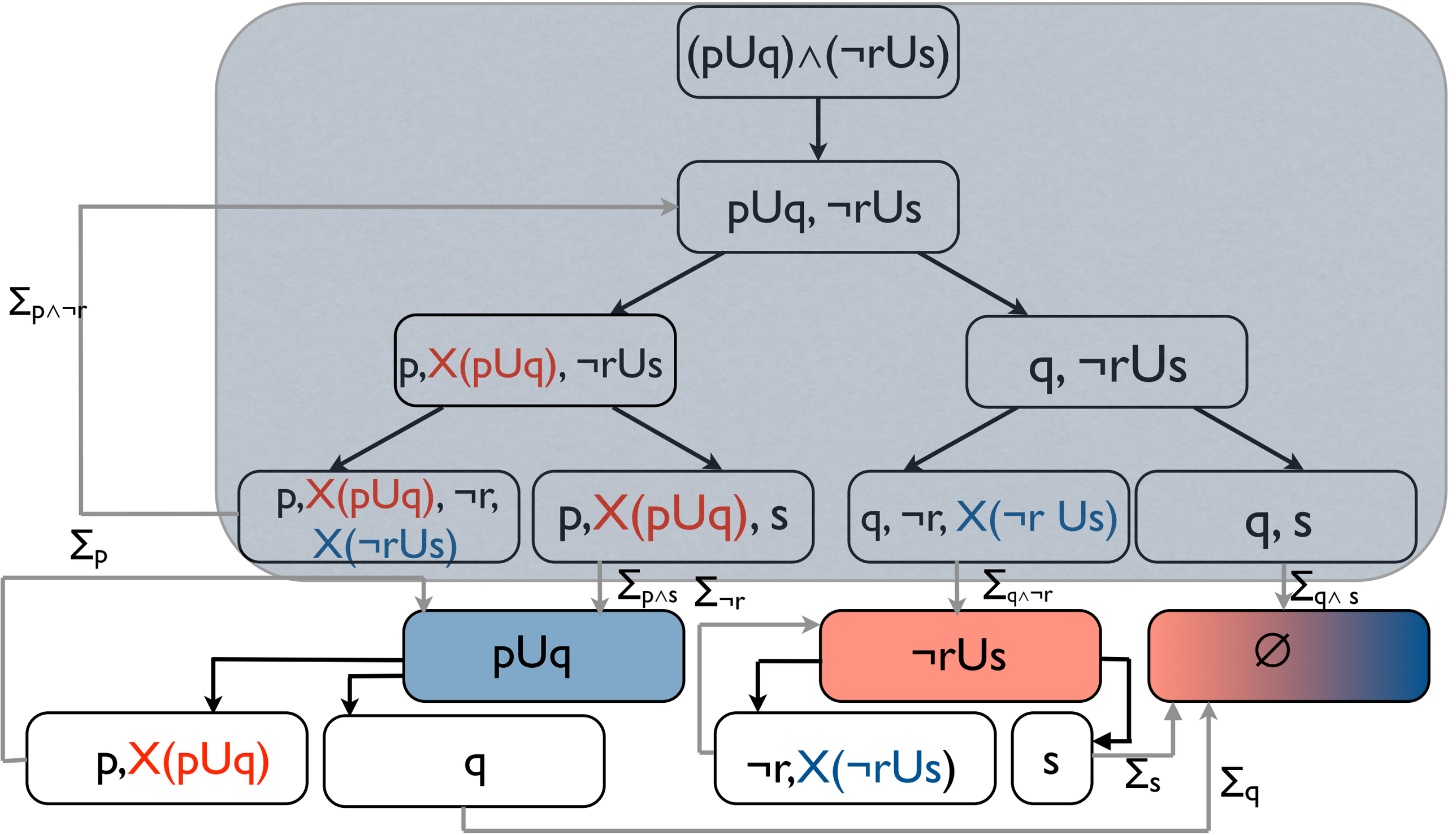
Exemple (suite)



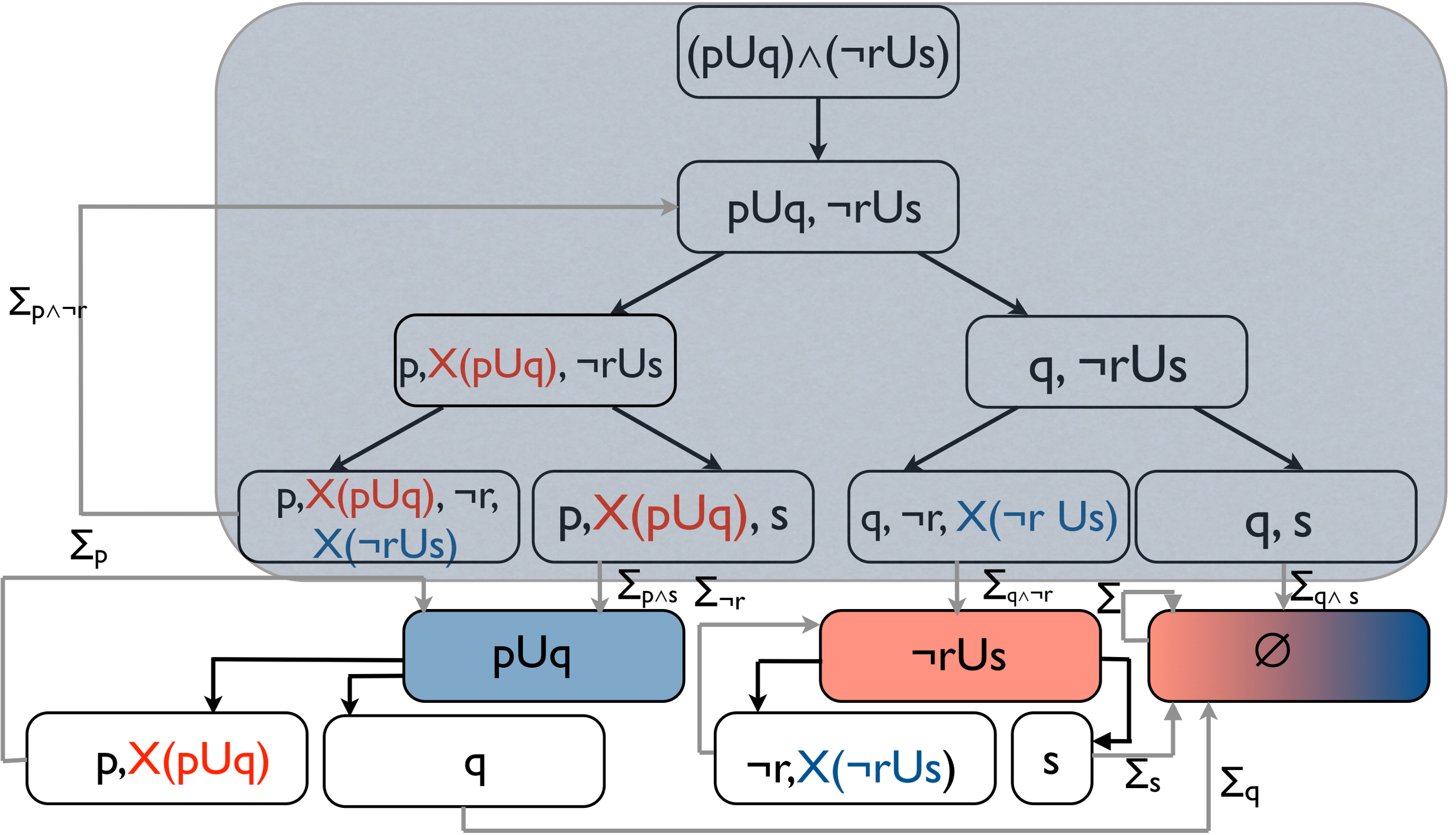
Exemple (suite)



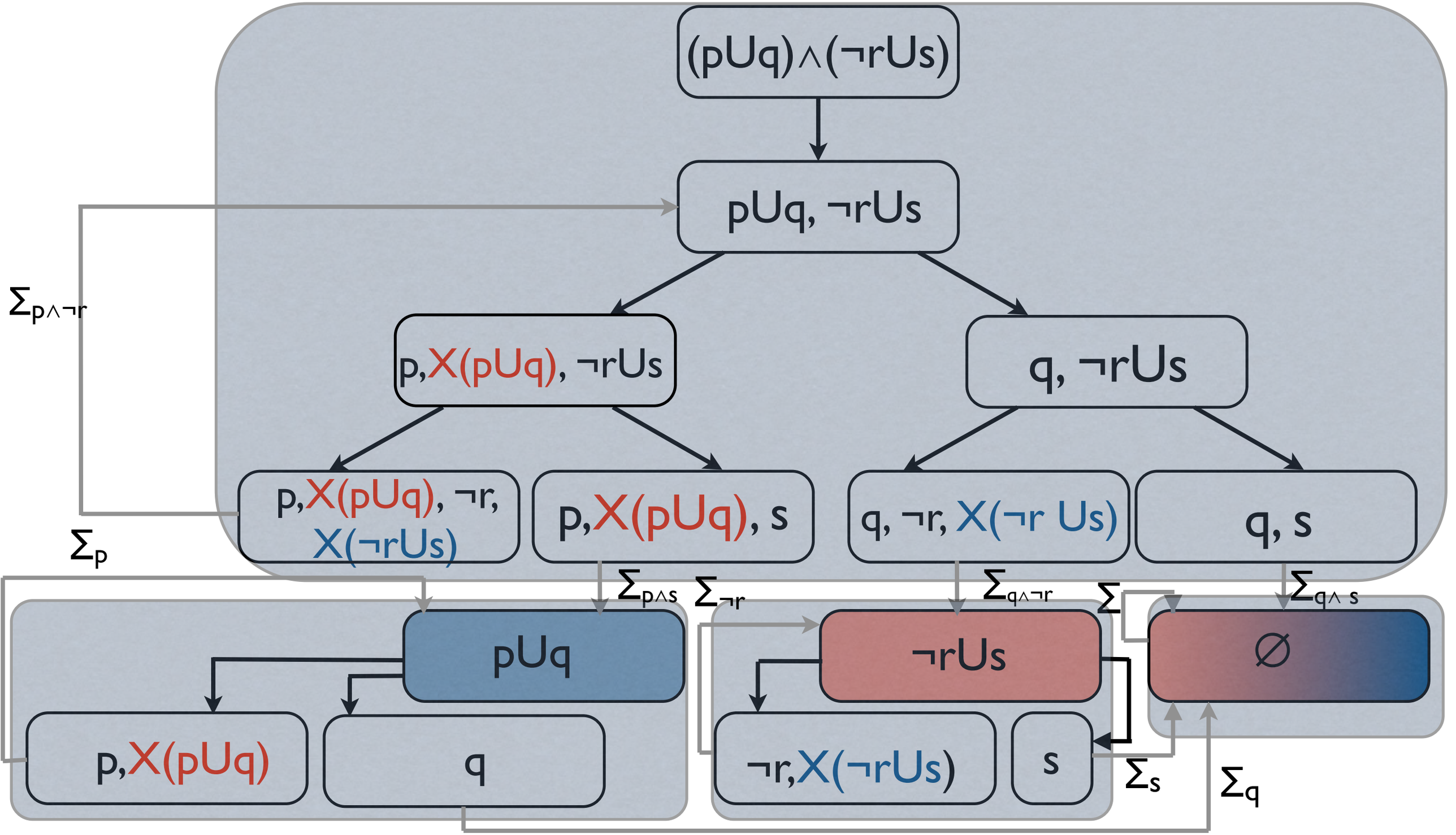
Exemple (suite)



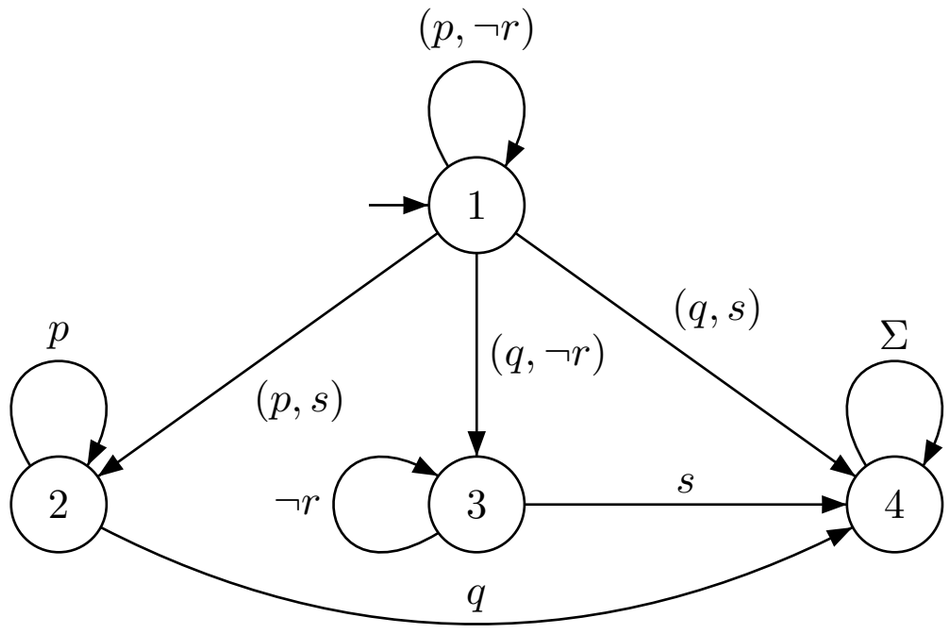
Exemple (suite)



Exemple (suite)



Exemple (fin)



$F_{pUq}=\{3,4\}$ $F_{\neg rUs}=\{2,4\}$

Construction - notations

- Si Z est un ensemble de formules réduit, on note next(Z) = $\{\psi | X\psi \in Z\}$, et on note Σ_Z = $\cap_{p \in Z} \Sigma_p \cap \cap_{\neg p \in Z} \Sigma_{\neg p}$

- Si Y se réduit en Z, on note $Y \dashrightarrow Z$, ou $Y \dashrightarrow_{!\varphi U \varphi'} Z$ pour la réduction $Z \cup \{\varphi U \varphi'\} \dashrightarrow Z \cup \{\varphi, X(\varphi U \varphi')\}$

Construction - notations (cont.)

- On note Red(Y)={Z réduit | Y⇢* Z} et on note $\mathrm{Red}_{\psi}(Y)$ = {Z réduit | Y⇢* Z sans utiliser de transition marquée par $!\psi$} pour tout ψ sous-formule de type Until.

Construction de l'automate A_φ

- Q= 2^{ψ}, ψ sous-formule de φ
- I = {φ}
- T= {(Y, a, next(Z))| Y ∈Q, Z∈Red(Y), a ∈Σ_Z }
- F={F_ψ|ψ sous-formule Until de φ} avec F_ψ={Y'| pour tout Z tel Y'=next(Z), il existe Y tq Z ∈ Red_ψ(Y)}

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Transformer M en un automate de Büchi

- Soit M=(Q,T,A, q_0, AP, l) une structure de Kripke. On construit un automate de Büchi B= (Q', Σ, q'0, T', F) tel que L(B)=⟦M⟧:
- Idée: on fait «basculer» les étiquettes des états vers les transitions + tous les états sont acceptants
  - $\Sigma=2^{AP}$
  - $Q'=T \cup \{q_0'\}$
  - $F=Q'$
  - Soit $t=(q_0,q)\in T$, alors $(q_0',l(q_0),t)\in T'$
  - Soient $t=(q,q')$ et $t'=(q',q'')\in T$, alors $(t,l(q'),t')\in T'$

Exemple

- au tableau

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Tester le vide de l'intersection

- Construire l'automate $A_M \otimes A_{\neg\varphi}$ tel que $L(A_M \otimes A_{\neg\varphi}) = L(A_M) \cap L(A_{\neg\varphi})$. (cf théorème)
- Rechercher s'il existe un mot accepté par $A_M \otimes A_{\neg\varphi}$. (cf théorème)

Model-Checking LTL: catching bugs with a lasso

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$. ✔

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔ O(|M|)
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$. ✔

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔ $O(|M|)$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔ $O(2^{|\varphi|})$
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$. ✔

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔ $O(|M|)$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔ $O(2^{|\varphi|})$
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$. ✔ $O(|M|.2^{|\varphi|})$

Model-Checking LTL: techniques à la volée

- Pas nécessaire de construire l'automate produit en entier
- On construit pas à pas, et on s'arrête lorsqu'on trouve un cycle (=contre-exemple).