Automates de Büchi

Benjamin Monmege

Motivation

Contexte :

- Programme : automates à propriétés $\mathcal{A}$, *processus Promela*

Motivation

Contexte :

- Programme : automates à propriétés $\mathcal{A}$, *processus Promela*
- Spécification : formule LTL φ, *processus* $\boldsymbol{never}$

Motivation

Contexte :

- Programme : automates à propriétés $\mathcal{A}$, *processus Promela*
- Spécification : formule LTL φ, *processus* $\mathtt{never}$

$$\mathcal{A} \models \varphi \text{ ?}$$

Motivation

Contexte :

- Programme : automates à propriétés $\mathcal{A}$, *processus Promela*
- Spécification : formule LTL φ, *processus* $\mathtt{never}$

$$\mathcal{A} \models \varphi \text{ ?}$$

i.e. toutes les exécutions maximales de $\mathcal{A}$ satisfont-elles φ ?

Motivation

Contexte :

- Programme : automates à propriétés $\mathcal{A}$, *processus Promela*
- Spécification : formule LTL φ, *processus* $\mathtt{never}$

$$\mathcal{A} \models \varphi \text{ ?}$$

i.e. toutes les exécutions maximales de $\mathcal{A}$ satisfont-elles φ ?

$\rightarrow$ SPIN fait ça de façon automatique.

Motivation

Contexte :

- Programme : automates à propriétés $\mathcal{A}$, *processus Promela*
- Spécification : formule LTL φ, *processus* $\mathtt{never}$

$$\mathcal{A} \models \varphi \text{ ?}$$

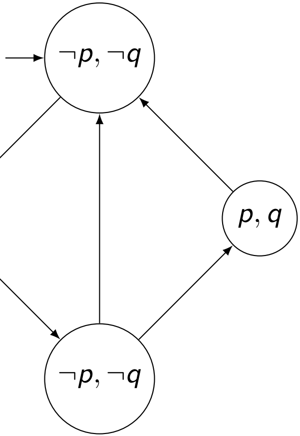
i.e. toutes les exécutions maximales de $\mathcal{A}$ satisfont-elles φ ?

$\rightarrow$ SPIN fait ça de façon automatique.

- Comment fait-il ?
- A quoi sert le processus never ?

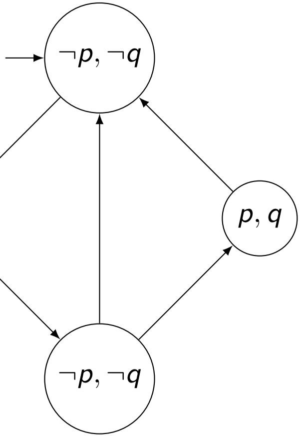
Motivation : exemple de la Démo 3

Programme :



Motivation : exemple de la Démo 3

Programme :

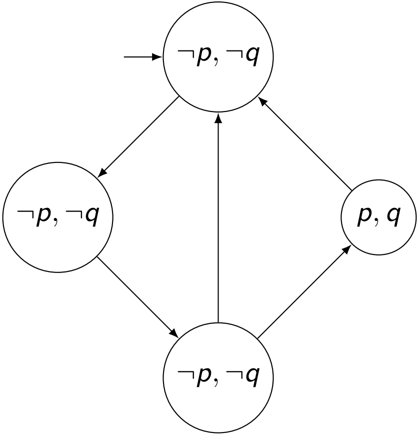


Spécification :

$$\varphi = G(p \Rightarrow XFq)$$

Motivation : exemple de la Démo 3

Programme :



Spécification :

$$\varphi = G(p \Rightarrow XFq)$$

On a vu que SPIN trouvait un contre-exemple...

Programme du jour

Automates de Büchi

- Définitions
- Exemples
- A propos de déterminisation

Programme du jour

Automates de Büchi

- Définitions
- Exemples
- A propos de déterminisation

LTL et automates de Büchi

- De LTL aux automates
- Réciproquement ?

Programme du jour

Automates de Büchi

- Définitions
- Exemples
- A propos de déterminisation

LTL et automates de Büchi

- De LTL aux automates
- Réciproquement ?

Model Checking de LTL

- Approche générale
- Tester le vide des automates de Büchi

Section 1

Automates de Büchi

Automates de Büchi

Définition (Automate de Büchi)

Un *automate de Büchi* $\mathcal{A}$ est un quintuplet $\mathcal{A} = (Q, \Sigma, T, I, F)$ où :

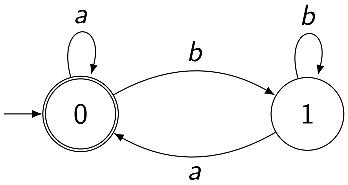
- Q est un ensemble fini d'états
- Σ est un alphabet (fini)
- $T \subseteq Q \times \Sigma \times Q$ est un ensemble (fini) de transitions
- $Q_i \subseteq Q$ est l'ensemble des états initiaux
- $Q_a \subseteq Q$ est l'ensemble des états **acceptants**

Automates de Büchi

Définition (Automate de Büchi)

Un *automate de Büchi* $\mathcal{A}$ est un quintuplet $\mathcal{A} = (Q, \Sigma, T, I, F)$ où :

- Q est un ensemble fini d'états
- Σ est un alphabet (fini)
- $T \subseteq Q \times \Sigma \times Q$ est un ensemble (fini) de transitions
- $Q_i \subseteq Q$ est l'ensemble des états initiaux
- $Q_a \subseteq Q$ est l'ensemble des états **acceptants**



Automates de Büchi

Modèles : mots infinis sur l'alphabet Σ (notation Σ^ω).

Automates de Büchi

Modèles : mots infinis sur l'alphabet Σ (notation Σ^ω).

Définition

Un *chemin* de $\mathcal{A}$ est une séquence de transitions consécutives : $t_1 \dots t_n$.
Son *étiquette* est le mot $a_1 \dots a_n$, où a_i est la lettre lue par t_i.

Automates de Büchi

Modèles : mots infinis sur l'alphabet Σ (notation Σ^ω).

Définition

Un *chemin* de $\mathcal{A}$ est une séquence de transitions consécutives : $t_1 \dots t_n$.
Son *étiquette* est le mot $a_1 \dots a_n$, où a_i est la lettre lue par t_i.

Un chemin est *acceptant* si :

- il part d'un état initial de Q_i
- il visite infiniment souvent un état acceptant de Q_a

Automates de Büchi

Modèles : mots infinis sur l'alphabet Σ (notation Σ^ω).

Définition

Un *chemin* de $\mathcal{A}$ est une séquence de transitions consécutives : $t_1 \dots t_n$.
Son *étiquette* est le mot $a_1 \dots a_n$, où a_i est la lettre lue par t_i.

Un chemin est *acceptant* si :

- il part d'un état initial de Q_i
- il visite infiniment souvent un état acceptant de Q_a

Un mot infini $w \in \Sigma^\omega$ est accepté par $\mathcal{A}$ ssi il existe un chemin acceptant de $\mathcal{A}$ d'étiquette w.

Automates de Büchi

Modèles : mots infinis sur l'alphabet Σ (notation Σ^ω).

Définition

Un *chemin* de $\mathcal{A}$ est une séquence de transitions consécutives : $t_1 \dots t_n$.
Son *étiquette* est le mot $a_1 \dots a_n$, où a_i est la lettre lue par t_i.

Un chemin est *acceptant* si :

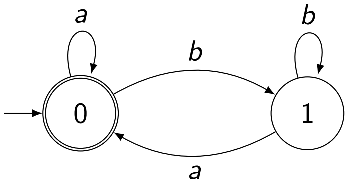
- il part d'un état initial de Q_i
- il visite infiniment souvent un état acceptant de Q_a

Un mot infini $w \in \Sigma^\omega$ est accepté par $\mathcal{A}$ ssi il existe un chemin acceptant de $\mathcal{A}$ d'étiquette w.

Définition

Le langage accepté/reconnu par $\mathcal{A}$, noté $L(\mathcal{A})$, est l'ensemble des mots $w \in \Sigma^\omega$ acceptés par $\mathcal{A}$.

Exemple



Définition

Le langage accepté/reconnu par $\mathcal{A}$, noté $L(\mathcal{A})$, est l'ensemble des mots $w \in \Sigma^\omega$ acceptés par $\mathcal{A}$.

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

$$\text{par exemple,} \qquad Prop = \{on, chaud, stop\}$$

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

$$\text{par exemple,} \qquad Prop = \{on, chaud, stop\}$$

$\Rightarrow$ Alphabet particulier : $\Sigma = 2^{Prop}$

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

par exemple, $Prop = \{on, chaud, stop\}$

$\Rightarrow$ Alphabet particulier : $\Sigma = 2^{Prop}$

Exemple :

- $Prop = \{p, q\}$
- $\Sigma = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

par exemple, $\quad Prop = \{on, chaud, stop\}$

$\Rightarrow$ Alphabet particulier : $\quad \Sigma = 2^{Prop}$

Exemple :

- $Prop = \{p, q\}$
- $\Sigma = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$

Ensemble de lettres : formules booléennes sur les propositions atomiques

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

$$\text{par exemple,} \qquad Prop = \{on, chaud, stop\}$$

$\Rightarrow$ Alphabet particulier : $\qquad \Sigma = 2^{Prop}$

Exemple :

- $Prop = \{p, q\}$
- $\Sigma = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$

Ensemble de lettres : formules booléennes sur les propositions atomiques

Exemples :

- p représente $\{\{p\}, \{p, q\}\}$

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

$$\text{par exemple,} \qquad Prop = \{on, chaud, stop\}$$

$\Rightarrow$ Alphabet particulier : $\qquad \Sigma = 2^{Prop}$

Exemple :

- $Prop = \{p, q\}$
- $\Sigma = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$

Ensemble de lettres : formules booléennes sur les propositions atomiques

Exemples :

- p représente $\{\{p\}, \{p, q\}\}$
- $p \wedge \neg q$ représente $\{\{p\}\}$

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

par exemple, $\qquad Prop = \{on, chaud, stop\}$

$\Rightarrow$ Alphabet particulier : $\qquad \Sigma = 2^{Prop}$

Exemple :

- $Prop = \{p, q\}$
- $\Sigma = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$

Ensemble de lettres : formules booléennes sur les propositions atomiques

Exemples :

- p représente $\{\{p\}, \{p, q\}\}$
- $p \wedge \neg q$ représente $\{\{p\}\}$
- $p \vee \neg q$ représente $\{\emptyset, \{p\}, \{p, q\}\}$

Propriétés atomiques

Dans notre contexte, on s'intéresse à des automates à propriétés :

$$\text{par exemple,} \qquad Prop = \{on, chaud, stop\}$$

$\Rightarrow$ Alphabet particulier : $\Sigma = 2^{Prop}$

Exemple :

- $Prop = \{p, q\}$
- $\Sigma = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$

Ensemble de lettres : formules booléennes sur les propositions atomiques

Exemples :

- p représente $\{\{p\}, \{p, q\}\}$
- $p \wedge \neg q$ représente $\{\{p\}\}$
- $p \vee \neg q$ représente $\{\emptyset, \{p\}, \{p, q\}\}$
- $\texttt{True}$ représente Σ

Exemples

A propos de déterminisation...

Définition (Déterminisme)

On dit que $\mathcal{A} = (Q, \Sigma, T, Q_i, Q_a)$ est *déterministe* si :

- $|Q_i| = 1$
- pour tout $p \in Q$ et $a \in \Sigma$, il existe au plus un état q tel que $(p, a, q) \in T$.

A propos de déterminisation...

Définition (Déterminisme)

On dit que $\mathcal{A} = (Q, \Sigma, T, Q_i, Q_a)$ est *déterministe* si :

- $|Q_i| = 1$
- pour tout $p \in Q$ et $a \in \Sigma$, il existe au plus un état q tel que $(p, a, q) \in T$.

Rappel : on peut toujours déterminiser un automate de mots finis.

A propos de déterminisation...

Définition (Déterminisme)

On dit que $\mathcal{A} = (Q, \Sigma, T, Q_i, Q_a)$ est *déterministe* si :

- $|Q_i| = 1$
- pour tout $p \in Q$ et $a \in \Sigma$, il existe au plus un état q tel que $(p, a, q) \in T$.

Rappel : on peut toujours déterminiser un automate de mots finis.

Attention : Ce n'est pas vrai pour les automates de Büchi !

A propos de déterminisation...

Lemme

Le langage $L = \{w \in \{a,b\}^\omega \mid |w|_a < +\infty\}$ *ne peut pas être reconnu par un automate de Büchi déterministe.*

A propos de déterminisation...

Lemme

Le langage $L = \{w \in \{a, b\}^\omega \mid |w|_a < +\infty\}$ *ne peut pas être reconnu par un automate de Büchi déterministe.*

Démonstration.

Par l'absurde : soit $\mathcal{A}$ un automate de Büchi déterministe reconnaissant L.

□

Section 2

LTL et automates de Büchi

LTL et automates de Büchi

Pour φ formule de LTL sur *Prop* (en posant $\Sigma = 2^{Prop}$) on note :

$$L_\varphi = \{w \in \Sigma^\omega \mid w \models \varphi\}$$

Théorème

Pour toute formule $\varphi \in LTL$, il existe un automate de Büchi $\mathcal{A}_\varphi$ tel que $L(\mathcal{A}_\varphi) = L_\varphi$.

LTL et automates de Büchi

Pour φ formule de LTL sur *Prop* (en posant $\Sigma = 2^{Prop}$) on note :

$$L_\varphi = \{w \in \Sigma^\omega \mid w \models \varphi\}$$

Théorème

Pour toute formule $\varphi \in LTL$, il existe un automate de Büchi $\mathcal{A}_\varphi$ tel que $L(\mathcal{A}_\varphi) = L_\varphi$.

Remarque : dans le pire cas, le nombre d'états de B_φ peut être exponentiel dans la taille de φ

LTL et automates de Büchi

Pour φ formule de LTL sur *Prop* (en posant $\Sigma = 2^{Prop}$) on note :

$$L_\varphi = \{w \in \Sigma^\omega \mid w \models \varphi\}$$

Théorème

Pour toute formule $\varphi \in LTL$, il existe un automate de Büchi $\mathcal{A}_\varphi$ tel que $L(\mathcal{A}_\varphi) = L_\varphi$.

Remarque : dans le pire cas, le nombre d'états de B_φ peut être exponentiel dans la taille de φ

Remarque : C'est ce que fait SPIN quand on utilise la commande

```
spin -f '[]<>(p-> Xq)'
```

LTL et automates de Büchi

Pour φ formule de LTL sur *Prop* (en posant $\Sigma = 2^{Prop}$) on note :

$$L_\varphi = \{w \in \Sigma^\omega \mid w \models \varphi\}$$

Théorème

Pour toute formule $\varphi \in LTL$, il existe un automate de Büchi $\mathcal{A}_\varphi$ tel que $L(\mathcal{A}_\varphi) = L_\varphi$.

Remarque : dans le pire cas, le nombre d'états de B_φ peut être exponentiel dans la taille de φ

Remarque : C'est ce que fait SPIN quand on utilise la commande

```
spin -f '[]<>(p-> Xq)'
```

La preuve du théorème se fait par induction sur les formules de LTL. On ne va pas faire tous les cas, ce serait trop technique.

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

$\varphi = p$

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

$\varphi = p$

$\varphi = \varphi_1 \vee \varphi_2$

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

$\varphi = p$

$\varphi = \varphi_1 \vee \varphi_2$

$\varphi = \varphi_1 \wedge \varphi_2$

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

$\varphi = p$

$\varphi = \varphi_1 \vee \varphi_2$

$\varphi = \varphi_1 \wedge \varphi_2$

$\varphi = X\varphi_1$

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

$\varphi = p$

$\varphi = \varphi_1 \vee \varphi_2$

$\varphi = \varphi_1 \wedge \varphi_2$

$\varphi = X\varphi_1$

$\varphi = F\varphi_1$

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

$\varphi = p$

$\varphi = \varphi_1 \vee \varphi_2$

$\varphi = \varphi_1 \wedge \varphi_2$

$\varphi = X\varphi_1$

$\varphi = F\varphi_1$

$\varphi = Gp$

De LTL aux automates de Büchi

Définition (LTL)

$\varphi ::= p \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid X\varphi \mid \varphi_1 U \varphi_2$

$\varphi = p$

$\varphi = \varphi_1 \vee \varphi_2$

$\varphi = \varphi_1 \wedge \varphi_2$

$\varphi = X\varphi_1$

$\varphi = F\varphi_1$

$\varphi = Gp$

$\varphi = pUq$

Exercice 1

Des automates de Büchi à LTL ?

Lemme

La logique LTL est strictement moins expressive que les automates de Büchi.

Des automates de Büchi à LTL ?

Lemme

La logique LTL est strictement moins expressive que les automates de Büchi.

Contre-exemple :

$$L = \{w \in \Sigma^\omega \mid |w|_b < +\infty \text{ et } |w|_b \equiv 0\ [2]\}$$

Des automates de Büchi à LTL ?

Lemme

La logique LTL est strictement moins expressive que les automates de Büchi.

Contre-exemple :

$$L = \{w \in \Sigma^\omega \mid |w|_b < +\infty \text{ et } |w|_b \equiv 0\ [2]\}$$

Théorème

Etant donné un automate de Büchi $\mathcal{A}$, on peut décider s'il existe une formule LTL φ telle que $L(\mathcal{A}) = L_\varphi$.

Section 3

Model Checking de LTL

Approche générale

Entrée : $\mathcal{P}$ automate à propriétés, $\varphi \in LTL$
Question : $\mathcal{P} \models \varphi$?

Approche générale

Entrée : $\mathcal{P}$ automate à propriétés, $\varphi \in LTL$
Question : $\mathcal{P} \models \varphi$?

Méthode : chercher une exécution de $\mathcal{P}$ qui ne satisfait pas φ

Approche générale

Entrée : $\mathcal{P}$ automate à propriétés, $\varphi \in LTL$
Question : $\mathcal{P} \models \varphi$?

Méthode : chercher une exécution de $\mathcal{P}$ qui ne satisfait pas φ

1. Construire l'automate de Büchi $\mathcal{A}_{\neg\varphi}$ pour la formule $\neg\varphi$

Approche générale

Entrée : $\mathcal{P}$ automate à propriétés, $\varphi \in LTL$
Question : $\mathcal{P} \models \varphi$?

Méthode : chercher une exécution de $\mathcal{P}$ qui ne satisfait pas φ

1. Construire l'automate de Büchi $\mathcal{A}_{\neg\varphi}$ pour la formule $\neg\varphi$
2. Construire l'automate de Büchi produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$

Approche générale

Entrée : $\mathcal{P}$ automate à propriétés, $\varphi \in LTL$
Question : $\mathcal{P} \models \varphi$?

Méthode : chercher une exécution de $\mathcal{P}$ qui ne satisfait pas φ

1. Construire l'automate de Büchi $\mathcal{A}_{\neg\varphi}$ pour la formule $\neg\varphi$
2. Construire l'automate de Büchi produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$
3. Tester si $L(\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}) = \emptyset$

Approche générale

Entrée : $\mathcal{P}$ automate à propriétés, $\varphi \in LTL$
Question : $\mathcal{P} \models \varphi$?

Méthode : chercher une exécution de $\mathcal{P}$ qui ne satisfait pas φ

1. Construire l'automate de Büchi $\mathcal{A}_{\neg\varphi}$ pour la formule $\neg\varphi$
2. Construire l'automate de Büchi produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$
3. Tester si $L(\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}) = \emptyset$

Si le langage est non-vide, on obtient un contre-exemple.

Approche générale : exemple du début

- Formule LTL : $\varphi = G(p \Rightarrow XFq) \quad \rightsquigarrow \quad \neg\varphi = F(p \wedge XG\neg q)$

Approche générale : exemple du début

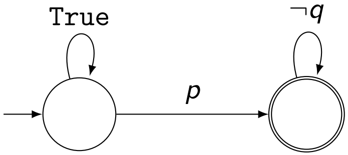
- Formule LTL : $\varphi = G(p \Rightarrow XFq) \quad \rightsquigarrow \quad \neg\varphi = F(p \wedge XG\neg q)$

$\mathcal{A}_{\neg\varphi}$:

Approche générale : exemple du début

- Formule LTL : $\varphi = G(p \Rightarrow XFq) \quad \rightsquigarrow \quad \neg\varphi = F(p \wedge XG\neg q)$

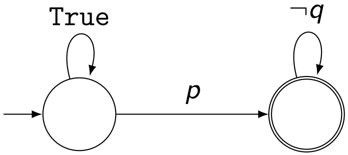
$\mathcal{A}_{\neg\varphi}$:



Approche générale : exemple du début

- Formule LTL : $\varphi = G(p \Rightarrow XFq) \quad \rightsquigarrow \quad \neg\varphi = F(p \wedge XG\neg q)$

$\mathcal{A}_{\neg\varphi}$:



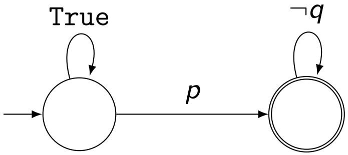
Construction du produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$:

- les états sont les paires d'états $(q_{\mathcal{P}}, q_{\mathcal{A}_{\neg\varphi}})$

Approche générale : exemple du début

- Formule LTL : $\varphi = G(p \Rightarrow XFq) \quad \rightsquigarrow \quad \neg\varphi = F(p \wedge XG\neg q)$

$\mathcal{A}_{\neg\varphi}$:



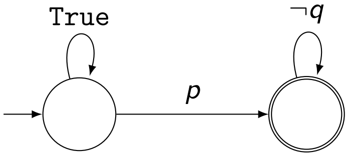
Construction du produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$:

- les états sont les paires d'états $(q_\mathcal{P}, q_{\mathcal{A}_{\neg\varphi}})$
- états initiaux : $(q_\mathcal{P}, q_{\mathcal{A}_{\neg\varphi}})$ avec $q_\mathcal{P}$ initial dans $\mathcal{P}$ et $q_{\mathcal{A}_{\neg\varphi}}$ initial dans $\mathcal{A}_{\neg\varphi}$

Approche générale : exemple du début

- Formule LTL : $\varphi = G(p \Rightarrow XFq) \quad \rightsquigarrow \quad \neg\varphi = F(p \wedge XG\neg q)$

$\mathcal{A}_{\neg\varphi}$:



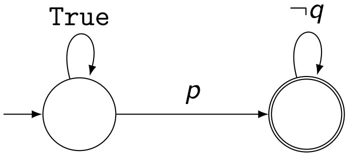
Construction du produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$:

- les états sont les paires d'états $(q_{\mathcal{P}}, q_{\mathcal{A}_{\neg\varphi}})$
- états initiaux : $(q_{\mathcal{P}}, q_{\mathcal{A}_{\neg\varphi}})$ avec $q_{\mathcal{P}}$ initial dans $\mathcal{P}$ et $q_{\mathcal{A}_{\neg\varphi}}$ initial dans $\mathcal{A}_{\neg\varphi}$
- états acceptants : $(q_{\mathcal{P}}, q_{\mathcal{A}_{\neg\varphi}})$ avec $q_{\mathcal{A}_{\neg\varphi}}$ acceptant dans $\mathcal{A}_{\neg\varphi}$

Approche générale : exemple du début

- Formule LTL : $\varphi = G(p \Rightarrow XFq) \quad \rightsquigarrow \quad \neg\varphi = F(p \wedge XG\neg q)$

$\mathcal{A}_{\neg\varphi}$:



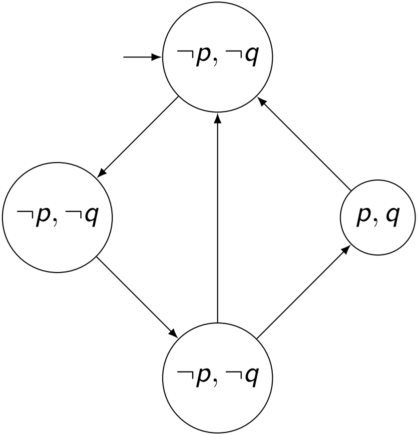
Construction du produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$:

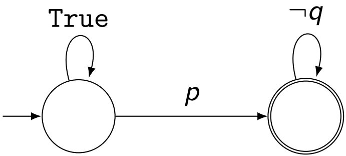
- les états sont les paires d'états $(q_\mathcal{P}, q_{\mathcal{A}_{\neg\varphi}})$
- états initiaux : $(q_\mathcal{P}, q_{\mathcal{A}_{\neg\varphi}})$ avec $q_\mathcal{P}$ initial dans $\mathcal{P}$ et $q_{\mathcal{A}_{\neg\varphi}}$ initial dans $\mathcal{A}_{\neg\varphi}$
- états acceptants : $(q_\mathcal{P}, q_{\mathcal{A}_{\neg\varphi}})$ avec $q_{\mathcal{A}_{\neg\varphi}}$ acceptant dans $\mathcal{A}_{\neg\varphi}$
- transitions : $(S \subseteq Prop)$

$$(q_\mathcal{P}, q_{\mathcal{A}_{\neg\varphi}}) \xrightarrow{S} (q'_\mathcal{P}, q'_{\mathcal{A}_{\neg\varphi}}) \iff \begin{cases} q_\mathcal{P} \rightarrow q'_\mathcal{P} \text{ dans } \mathcal{P} \\ \text{les prop. de } S \text{ sont vraies dans } q_\mathcal{P} \\ q_{\mathcal{A}_{\neg\varphi}} \xrightarrow{S} q'_{\mathcal{A}_{\neg\varphi}} \text{ dans } \mathcal{A}_{\neg\varphi} \end{cases}$$

Approche générale : exemple du début (suite)

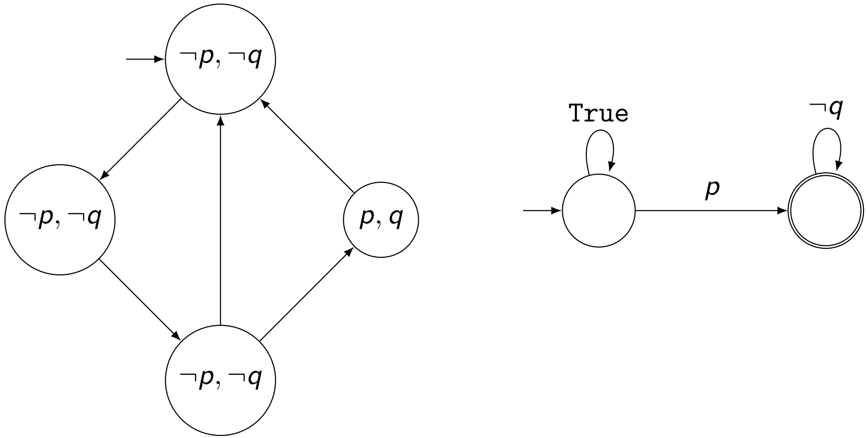
Produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$:





Approche générale : exemple du début (suite)

Produit $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$:



$\Rightarrow$ Il existe une séquence infinie acceptante, par exemple :

$$\emptyset\ \emptyset\ \emptyset\ \{p, q\}\ \emptyset\ \emptyset\ \emptyset\ \emptyset\ \emptyset \dots$$

Tester le vide d'un automate de Büchi

Comment tester qu'un automate de Büchi (par exemple $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$) accepte au moins un mot infini ?

Tester le vide d'un automate de Büchi

Comment tester qu'un automate de Büchi (par exemple $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$) accepte au moins un mot infini ?

Affirmation :
$L(A) \neq \emptyset \iff$ Il existe un chemin d'un état initial à un état acceptant, et une boucle autour de cet état acceptant.

Tester le vide d'un automate de Büchi

Comment tester qu'un automate de Büchi (par exemple $\mathcal{P} \otimes \mathcal{A}_{\neg\varphi}$) accepte au moins un mot infini ?

Affirmation :
$L(A) \neq \emptyset \iff$ Il existe un chemin d'un état initial à un état acceptant, et une boucle autour de cet état acceptant.

On cherche donc une composante fortement connexe C

- accessible depuis un état initial
- contenant un état acceptant
- non triviale (contenant au moins une transition)

Tester le vide d'un automate de Büchi (2)

Calcul des composantes fortement connexes, par exemple avec l'algorithme de Kosaraju :

Tester le vide d'un automate de Büchi (2)

Calcul des composantes fortement connexes, par exemple avec l'algorithme de Kosaraju :

1. Parcours en profondeur en partant des états initiaux, ce qui donne un ordre de sortie sur les sommets.
2. Dans cet ordre en sens décroissant, exécuter un deuxième parcours en profondeur itéré dans le graphe $\overleftarrow{G}$ obtenu en inversant les flèches. Chaque parcours élémentaire produit une composante fortement connexe.

Tester le vide d'un automate de Büchi (2)

Calcul des composantes fortement connexes, par exemple avec l'algorithme de Kosaraju :

1. Parcours en profondeur en partant des états initiaux, ce qui donne un ordre de sortie sur les sommets.
2. Dans cet ordre en sens décroissant, exécuter un deuxième parcours en profondeur itéré dans le graphe $\overleftarrow{G}$ obtenu en inversant les flèches. Chaque parcours élémentaire produit une composante fortement connexe.

Complexité : quadratique dans le nombre de sommets de G.

Tester le vide d'un automate de Büchi (2)

Calcul des composantes fortement connexes, par exemple avec l'algorithme de Kosaraju :

1. Parcours en profondeur en partant des états initiaux, ce qui donne un ordre de sortie sur les sommets.
2. Dans cet ordre en sens décroissant, exécuter un deuxième parcours en profondeur itéré dans le graphe $\overleftarrow{G}$ obtenu en inversant les flèches. Chaque parcours élémentaire produit une composante fortement connexe.

Complexité : quadratique dans le nombre de sommets de G.

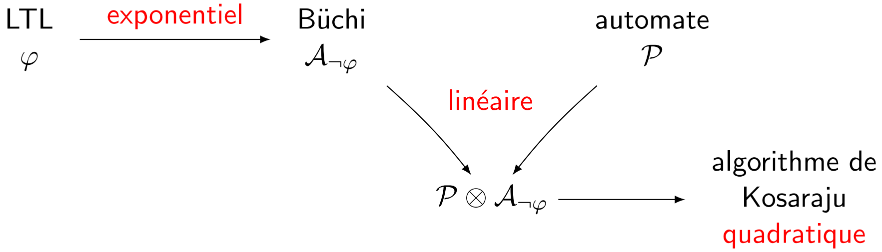
Exemple...

Exercice 2

Model Checking de LTL : résumé

$$
\begin{tikzcd}
\begin{array}{c}\text{LTL}\\ \varphi\end{array} \arrow[r, "\text{exponentiel}"] & \begin{array}{c}\text{Büchi}\\ \mathcal{A}_{\neg\varphi}\end{array} \arrow[dr, "\text{linéaire}"] & & \begin{array}{c}\text{automate}\\ \mathcal{P}\end{array} \arrow[dl] \\
& & \mathcal{P} \otimes \mathcal{A}_{\neg\varphi} \arrow[r] & \begin{array}{c}\text{algorithme de}\\ \text{Kosaraju}\\ \text{quadratique}\end{array}
\end{tikzcd}
$$

Model Checking de LTL : résumé



$\Longrightarrow$ Vérification de la propriété φ sur l'automate à propriétés $\mathcal{P}$ en temps

- exponentiel en la taille de φ (généralement petit)
- et quadratique en le nombre d'états de $\mathcal{P}$ (généralement grand)

Exercice 3

Nous avons vu aujourd'hui...

Automates de Büchi

- Définitions
- Exemples
- A propos de déterminisation

LTL et automates de Büchi

- De LTL aux automates
- Réciproquement ?

Model Checking de LTL

- Approche générale
- Tester le vide des automates de Büchi