

# Cinquième partie

## $TLA^+$ – la logique



# Plan

- 1 LTL et TLA<sup>+</sup>
  - Logique TLA<sup>+</sup>
  - Raffinage
- 2 Preuve axiomatique
- 3 Vérification de modèles



# La logique TLA<sup>+</sup>

## Expressions logiques

Expressions de LTL avec  $\Box$ ,  $\Diamond$ ,  $\leadsto$  (leads-to) et variables primées  
+ quantificateurs  $\forall, \exists$ .

Pas de  $\mathcal{U}$ , ni de  $\mathcal{W}$ , mais :

$$\Box(p \Rightarrow (p\mathcal{W}q)) = \Box(p \Rightarrow (p' \vee q))$$

$$\Box(p \Rightarrow (p\mathcal{U}q)) = \Box(p \Rightarrow (p' \vee q)) \wedge \Box(p \Rightarrow \Diamond q)$$



Équité / *Fairness*

## ENABLED

ENABLED  $\mathcal{A}$  est la fonction d'état qui est vraie dans l'état  $s$  ssi il existe un état  $t$  accessible depuis  $s$  par l'action  $\mathcal{A}$ .

*Weak/Strong Fairness*

- $WF_e(\mathcal{A}) \triangleq \Box\Diamond\neg(\text{ENABLED } \langle\mathcal{A}\rangle_e) \vee \Box\Diamond\langle\mathcal{A}\rangle_e$   
si  $\mathcal{A}$  est constamment déclenchable, elle sera déclenchée.
- $SF_e(\mathcal{A}) \triangleq \Diamond\Box\neg(\text{ENABLED } \langle\mathcal{A}\rangle_e) \vee \Box\Diamond\langle\mathcal{A}\rangle_e$   
si  $\mathcal{A}$  est infiniment souvent déclenchable, elle sera déclenchée.



# Forme d'une spécification TLA<sup>+</sup>

En général, une spécification TLA<sup>+</sup> est une conjonction

$$\mathcal{I} \wedge \Box[\mathcal{N}]_v \wedge \mathcal{E}$$

- $\mathcal{I}$  = prédicat d'état décrivant les états initiaux
- $\mathcal{N}$  = disjonction d'actions  $\mathcal{A}_1 \vee \mathcal{A}_2 \vee \mathcal{A}_3 \vee \dots$
- $\mathcal{E}$  = conjonction de contraintes d'équité portant sur les actions :  $WF_v(\mathcal{A}_1) \wedge SF_v(\mathcal{A}_3) \wedge \dots$



# Raffinage de spécification

## Raffinage simple

Une spécification (concrète)  $P_c$  raffine une spécification (abstraite)  $P_a$  si  $P_c \Rightarrow P_a$  : tout ce que fait  $P_c$  est possible dans  $P_a$ .

Cela signifie que si  $P_a \models P$  pour une propriété LTL quelconque, alors  $P_c \models P$ .



## Raffinage – exemple

### Somme abstraite

MODULE *somme1*

EXTENDS *Naturals*

CONSTANT *N*

VARIABLE *res*

*TypeInvariant*  $\triangleq res \in Nat$

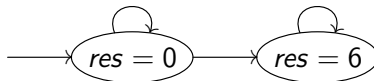
*Init*  $\triangleq res = 0$

*Next*  $\triangleq res' = ((N + 1) * N) \div 2$

*Spec*  $\triangleq Init \wedge \Box[Next]_{res} \wedge WF_{res}(Next)$

# Raffinage – exemple

Graphe des exécutions pour  $N = 3$





## Raffinage – exemple

### Somme plus concrète

MODULE *somme2*

EXTENDS *Naturals*

CONSTANT *N*

VARIABLE *res*, *acc*, *disp*

*TypeInvariant*  $\triangleq res \in Nat \wedge acc \in Nat \wedge disp \in SUBSET\ 1..N$

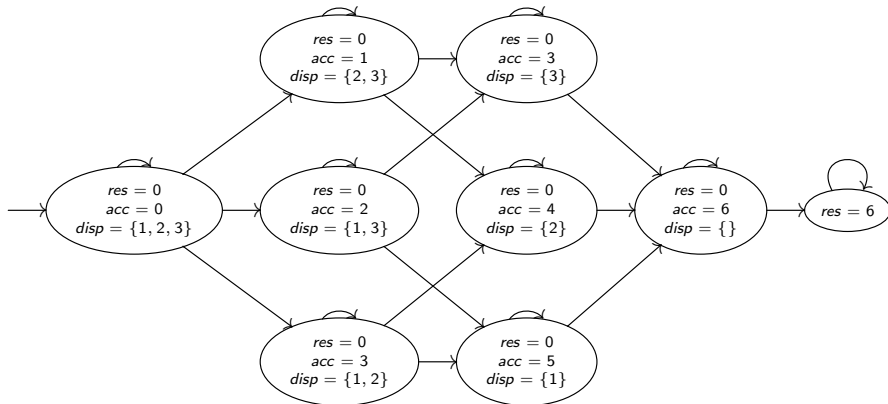
*Init*  $\triangleq res = 0 \wedge acc = 0 \wedge disp = 1..N$

*Next*  $\triangleq \forall \exists i \in disp : acc' = acc + i \wedge disp' = disp \setminus \{i\}$   
 $\wedge UNCHANGED\ res$

$\vee disp = \{\} \wedge res' = acc \wedge UNCHANGED\ \langle disp, acc \rangle$   
*Spec*  $\triangleq Init \wedge \Box [Next]_{\langle res, disp, acc \rangle} \wedge WF_{\langle res, disp, acc \rangle}(Next)$

# Raffinage – exemple

Graphe des exécutions pour  $N = 3$



Décomposition : introduction de transitions intermédiaires.

NT

# Raffinage – exemple

## Somme2 raffine somme1

```
MODULE somme2_raffine_somme1  
  
EXTENDS somme2  
  
Orig  $\triangleq$  INSTANCE somme1  
Raffinement  $\triangleq$  Orig! Spec  
  
THEOREM Spec  $\Rightarrow$  Orig! Spec
```



## Raffinage – exemple

### Somme concrète

MODULE *somme3*

EXTENDS *Naturals*

CONSTANT *N*

VARIABLE *res, acc, i*

*TypeInvariant*  $\triangleq res \in Nat \wedge acc \in Nat \wedge i \in 1 .. N$

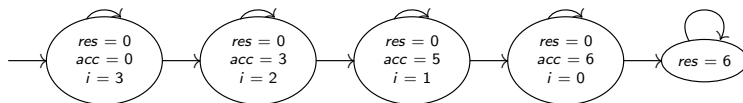
*Init*  $\triangleq res = 0 \wedge acc = 0 \wedge i = N$

*Next*  $\triangleq \vee i > 0 \wedge acc' = acc + i \wedge i' = i - 1 \wedge \text{UNCHANGED } res$   
 $\vee i = 0 \wedge res' = acc \wedge \text{UNCHANGED } \langle i, acc \rangle$

*Spec*  $\triangleq Init \wedge \Box[Next]_{\langle res, i, acc \rangle} \wedge WF_{\langle res, i, acc \rangle}(Next)$

# Raffinage – exemple

Graphe des exécutions pour  $N = 3$



Réduction du non-déterminisme + changement de représentation  
(raffinement de données)  $disp = 1..i$



## Raffinage – exemple

### Somme3 raffine somme2

```
MODULE somme3_raffine_somme2
EXTENDS somme3
dispMapping  $\triangleq$  1 .. i
Orig  $\triangleq$  INSTANCE somme2 WITH disp  $\leftarrow$  dispMapping
Raffinement  $\triangleq$  Orig! Spec
THEOREM Spec  $\Rightarrow$  Orig! Spec
```



# Plan

- 1 LTL et TLA<sup>+</sup>
  - Logique TLA<sup>+</sup>
  - Raffinage
- 2 Preuve axiomatique
- 3 Vérification de modèles



## Règles de preuve – simple temporal logic

$$\frac{F \text{ prouvable en } \text{logique propositionnelle}}{\Box F} \text{STL1}$$

$$\frac{F \Rightarrow G}{\Box F \Rightarrow \Box G} \text{STL4}$$

$$\overline{\Box F \Rightarrow F} \text{STL2}$$

$$\overline{\Box \Box F = \Box F} \text{STL3}$$

$$\overline{\Box(F \wedge G) = (\Box F) \wedge (\Box G)} \text{STL5}$$

$$\overline{\Diamond \Box F \wedge \Diamond \Box G = \Diamond \Box(F \wedge G)} \text{STL6}$$





## Règles de preuve – TLA<sup>+</sup> invariant

$$\frac{P \wedge (v' = v) \Rightarrow P'}{\Box P = P \wedge \Box [P \Rightarrow P']_v} \text{TLA1} \quad \frac{P \wedge [\mathcal{A}]_{v_1} \Rightarrow Q \wedge [\mathcal{B}]_{v_2}}{\Box P \wedge \Box [\mathcal{A}]_{v_1} \Rightarrow \Box Q \wedge \Box [\mathcal{B}]_{v_2}} \text{TLA2}$$

$$\frac{I \wedge [\mathcal{N}]_v \Rightarrow I'}{I \wedge \Box [\mathcal{N}]_v \Rightarrow \Box I} \text{INV1} \quad \frac{}{\Box I \Rightarrow (\Box [\mathcal{N}]_v = \Box [\mathcal{N} \wedge I \wedge I']_v)} \text{INV2}$$



## Règles de preuve – TLA<sup>+</sup> vivacité

$$\begin{array}{c}
 P \wedge [\mathcal{N}]_v \Rightarrow (P' \vee Q') \\
 P \wedge \langle \mathcal{N} \wedge \mathcal{A} \rangle_v \Rightarrow Q' \\
 P \Rightarrow \text{ENABLED } \langle \mathcal{A} \rangle_v \\
 \hline
 \Box[\mathcal{N}]_v \wedge WF_v(\mathcal{A}) \Rightarrow (P \leadsto Q) \text{ WF1}
 \end{array}$$

$$\begin{array}{c}
 P \wedge [\mathcal{N}]_v \Rightarrow (P' \vee Q') \\
 P \wedge \langle \mathcal{N} \wedge \mathcal{A} \rangle_v \Rightarrow Q' \\
 \Box P \wedge \Box[\mathcal{N}]_v \wedge \Box F \Rightarrow \Diamond \text{ENABLED } \langle \mathcal{A} \rangle_v \\
 \hline
 \Box[\mathcal{N}]_v \wedge SF_v(\mathcal{A}) \wedge \Box F \Rightarrow (P \leadsto Q) \text{ SF1}
 \end{array}$$



## Règles de preuve dérivées

$$\frac{\Box(P \Rightarrow \Box P) \wedge \Diamond P}{\Diamond \Box P} \text{LDSTBL}$$

$$\frac{P \rightsquigarrow Q \wedge Q \rightsquigarrow R}{P \rightsquigarrow R} \text{TRANS}$$

$$\frac{\forall m \in W : (P(m) \rightsquigarrow Q)}{(\exists m \in W : P(m)) \rightsquigarrow Q} \text{INFDIJ}$$



# Plan

- 1 LTL et TLA<sup>+</sup>
  - Logique TLA<sup>+</sup>
  - Raffinage
- 2 Preuve axiomatique
- 3 Vérification de modèles



# Vérification de modèles

## Principe

Construire le graphe des exécutions et étudier la propriété.

- $\Box P$ , où  $P$  est un prédicat d'état (sans variable primée) : au fur et à mesure de la construction des états.
- $\Box P(v, v')$ , où  $P(v, v')$  est un prédicat de transition (prédicat non temporel avec variables primées et non primées) : au fur et à mesure du calcul des transitions.
- Vivacité  $\Diamond P, P \rightsquigarrow Q \dots$  : une fois le graphe construit, chercher un cycle qui respecte les contraintes d'équité et qui invalide la propriété.

Uniquement sur des modèles finis, et, pratiquement, de petites tailles.



# Complexité

Soit  $|S|$  le nombre d'états d'un système  $\mathcal{S} = \langle S, I, R \rangle$  et  $|F|$  la taille (le nombre d'opérateurs temporels) d'une formule LTL  $F$ . La complexité en temps (et espace) pour vérifier  $\mathcal{S} \models F$  est  $O(|S| \times 2^{|F|})$ .



# Vérificateur TLC

Le vérificateur de modèles TLC sait vérifier :

- les spécifications avec des actions gardées ;
- (efficacement) les invariants sans variables primées :  $\Box P$  où  $P$  est un prédicat d'état ;
- les formules de sûreté pure avec variables primées et bégaiement :  $\Box[P]_v$  où  $P$  est un prédicat de transition ;
- $P \leadsto Q$  où  $P$  et  $Q$  sont des prédicats d'état (*sans* variables primées) ;
- les formules combinant  $\Box, \Diamond$  *sans* variables primées.

Note : l'espace d'états du système et des formules doit être fini : toute quantification bornée par exemple.

