STAGE MIM2

Magistère d'Informatique et de Modélisation

ENS Lyon - janvier/mars 2002

Laure Danthony

Vérification de systèmes temps-réel par instrumentation de code

PLAN :

- Présentation et cadre de l'étude
- Le langage IF - premiers exemples
- Vers une logique comportementale en IF
- Conclusion et perspectives

Présentation et cadre de l'étude

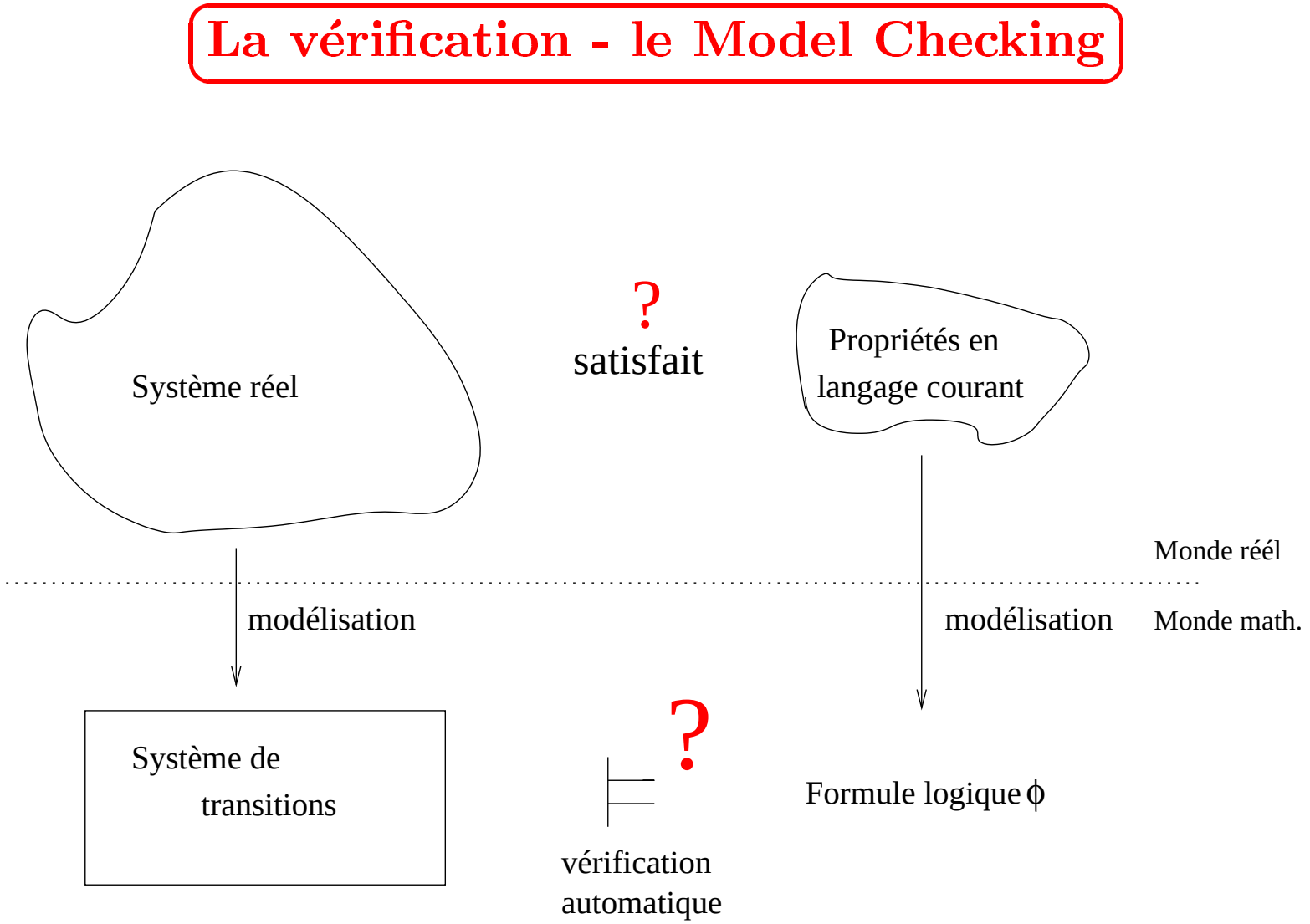

FIG. 1 – Schéma usuel de Model Checking

Les systèmes temps-réel

Système temps-réel = programme + plate forme (système dynamique : le temps fait partie du système)

⇒ introduction du temps :

- deux dimensions dépendantes : action + temps
- un temps global
- un temps qui diverge

Ambition

Trouver des modèles et des façons d'exprimer des propriétés temporisées qui permettent la vérification efficace de tels systèmes

Au Vérimag, l'équipe Kronos travaille sur de tels sujets

Le langage IF - premiers exemples

Les automates temporisés

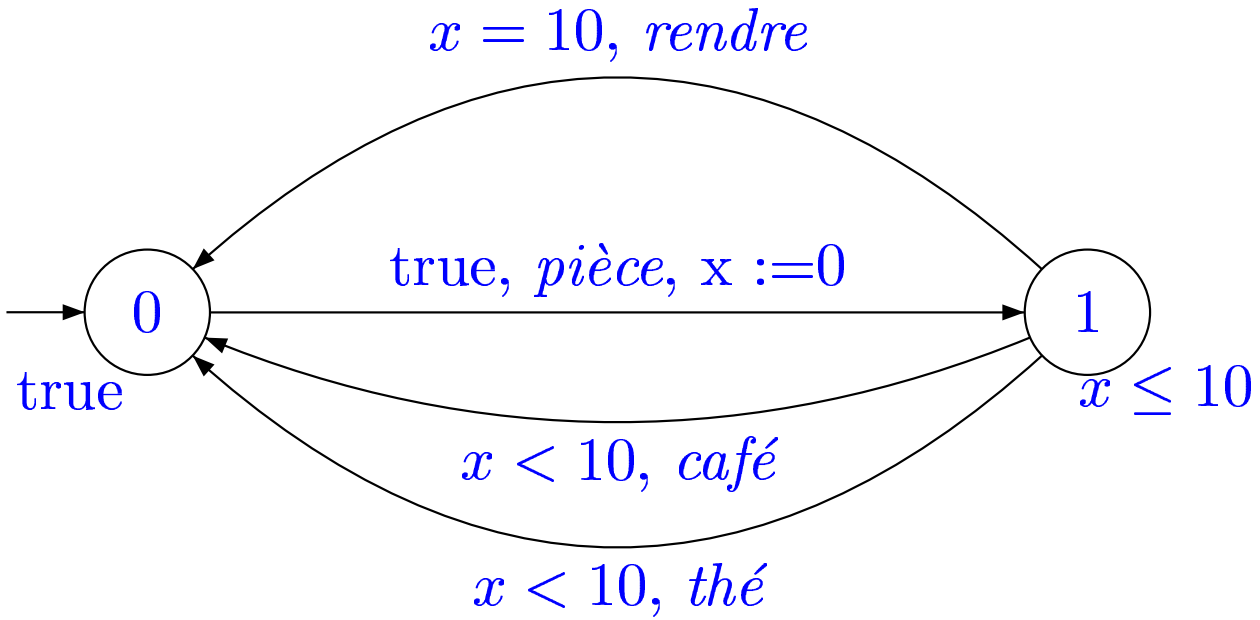

FIG. 2 – La machine à café/thé

Un peu d'histoire

IF = automates temporisés communiquant par canaux.

Langage maison à sémantique bien définie.

Possibilités :

- plusieurs processus qui tournent en parallèle
- variables partagées
- canaux de communication avec ou sans perte
- possibilité d'inclure du code externe
- ...

Autour de IF

Outils existants :

- Des traducteurs vers ou à partir de IF : SDL2IF, IF2C, ...
- Des outils de vérification : if.open (simulator, generator, evaluator), ...

Vers une logique comportementale en IF

Inconvénients des logiques temporelles

Logiques temporelles : CTL, TCTL, ...

Par exemple : `AG (start => EF_7 stop)`

Mais :

- Lourdeur des formules ;
- Difficulté d'obtenir les formules que l'on veut ;
- Difficulté de vérifier la cohérence d'ensembles de formules.

La notion d'automate observateur

On code un "sous-comportement", c'est-à-dire une abstraction du code.

Par exemple :

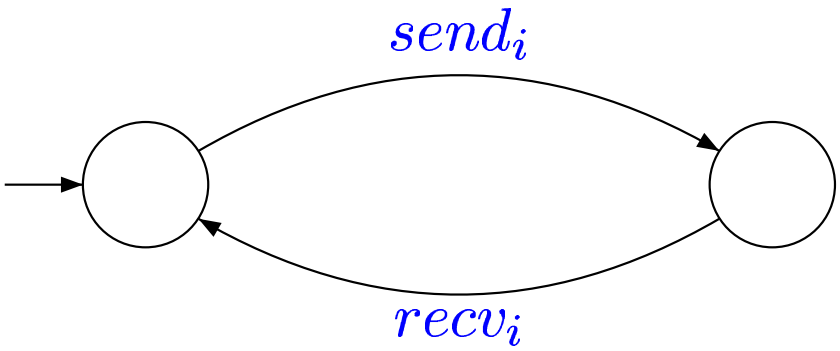

FIG. 3 – Description comportementale

$\Rightarrow$ C'est plus pratique !

Réalisation pratique en IF et vérification

Etapes :

- observateur = nouveau processus IF
- ajout éventuel de nouvelles variables et horloges
- mise en évidence d'un "mauvais état"
  $\Rightarrow$ "instrumentation"
- vérification de la non-atteignabilité de cet état à l'aide de `evaluator`

Propriétés courantes pour les protocoles

Exclusion mutuelle, send/receive, ...

$\Rightarrow$ Trois principaux types de propriétés :

- $\forall n, d_{min} \leq occ_{n+1}(e) - occ_n(e) \leq d_{max}(1)$
- $\forall n, d_{min} \leq occ_n(e_2) - occ_n(e_1) \leq d_{max}(2)$
- jitter :

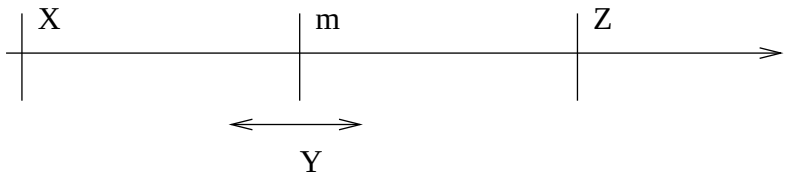

FIG. 4 – Erreur sur Y

Mise en œuvre de l'instrumentation de IF - succession

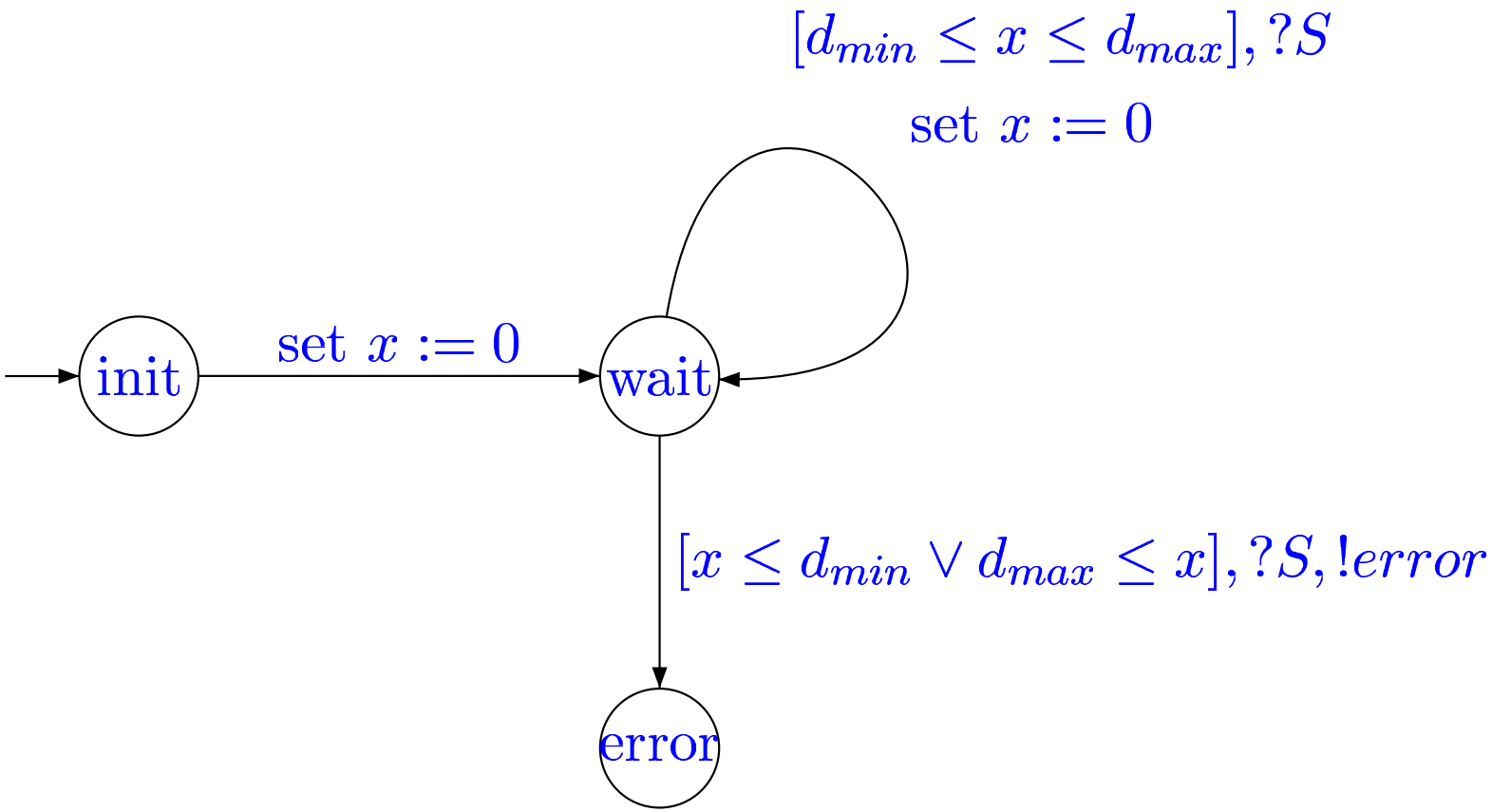

FIG. 5 – Propriété (1)

Mise en œuvre de l'instrumentation de IF - entrelacement

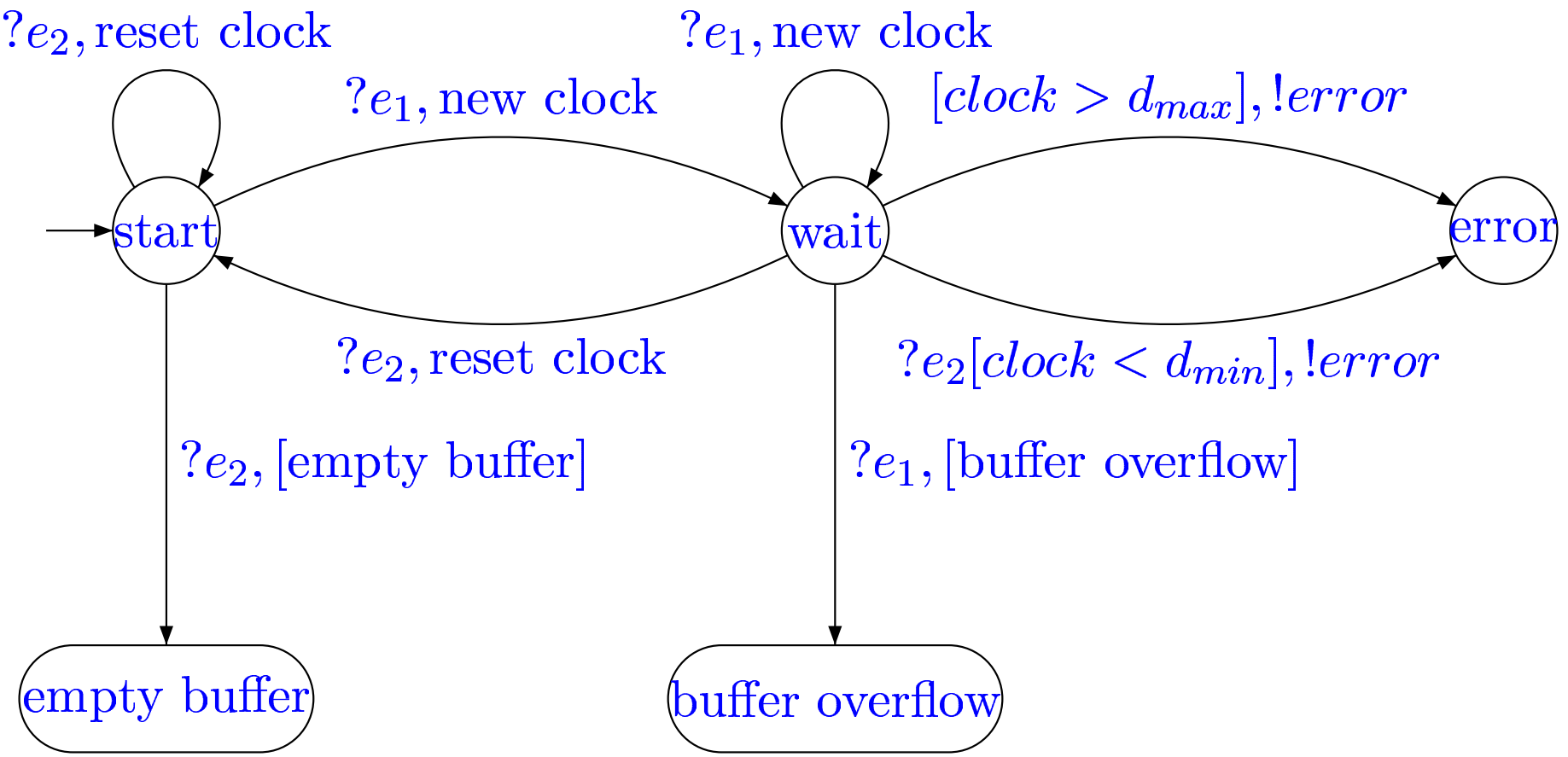

FIG. 6 – Propriété (2)- idée

clock : l'horloge *la plus ancienne*

Conclusion et perspectives

Bilan

Un stage riche en découvertes :

- Systèmes temps-réel - modélisation et difficultés
- L'outil IF - ce qui gravite autour
- La notion d'automate observateur - sa réalisation en IF

Prolongements possibles

Travail à suivre :

- Automatisation de l'instrumentation de IF pour les propriétés données.
- Etude de la possibilité de construire des observateurs pour d'autres types de formules.
- Etude de classes de formules : leur vérification est-elle réalisable pour IF avec les observateurs ?
- Accès au temps : intervalles (IF) ou avec une primitive `now()` ?

Remerciements

- Laboratoire Vérimag
- Joseph Sifakis

Enfin

Vous pouvez retrouver ce rapport de stage ainsi que ces transparents sur la page :

http://www.ens-lyon.fr/~ldanthon/verification/