printemps des sciences

Science et ville 14-20 mars 2005

ULB

Vérification de systèmes distribués

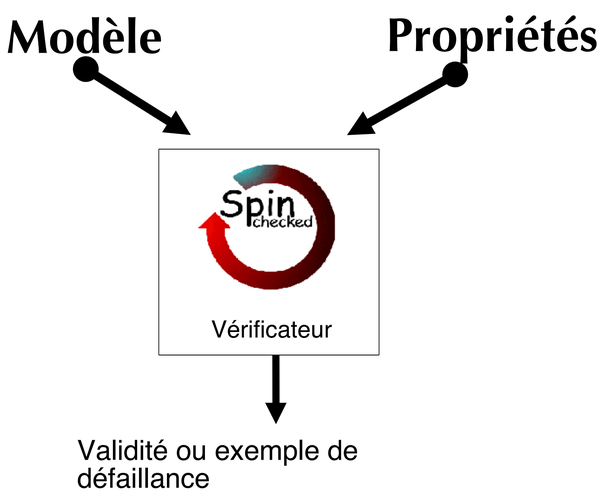
DÉPARTEMENT D'INFORMATIQUE

David Czerwonogora – Henry Ledoux – Olivier Godart

Pourquoi vérifier?

Dans un bon nombre d'applications, des logiciels informatiques prennent un rôle critique. Cela signifie que la moindre défaillance de ceux-ci peut provoquer des catastrophes. Qu'il s'agisse du contrôle de la signalisation des chemins de fer, des commandes d'aiguillages, ou d'une chaîne de fabrication dans une usine, nous voulons ne rien laisser au hasard, et être sûrs qu'aucun problème ne puisse survenir, même s'il y a très peu de chance que cela arrive.

Qu'est-ce que la vérification?



La *vérification* consiste à prouver sur base d'un modèle formel que certaines propriétés sont respectées, quel que soit l'ordre des événements. Cette tâche peut prendre des proportions énormes étant donné le nombre élevé de comportements différents que peut prendre un programme, même si celui-ci est petit. Un programme a en général un nombre infini d'exécutions possibles, mais on peut voir celui-ci comme ayant un nombre fini d'états. Il est ainsi possible de faire cette vérification à l'aide d'un ordinateur.

L'exclusion mutuelle

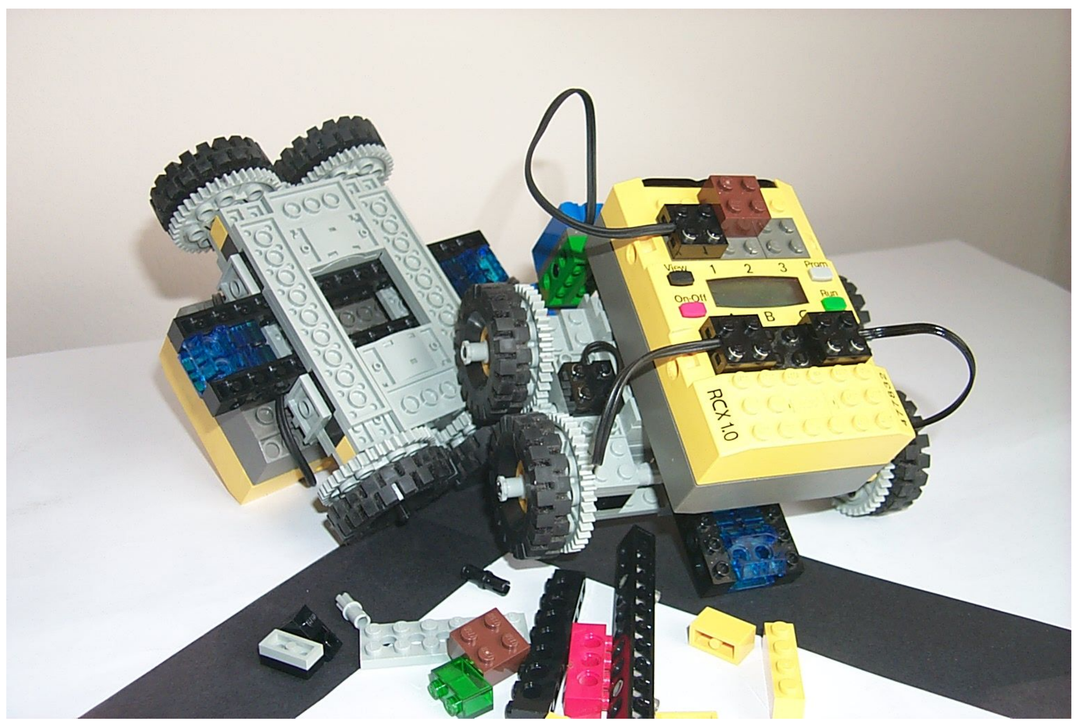
Le problème de l'exclusion mutuelle trouve sa place partout ; il se présente comme suit :

Plusieurs processus veulent accéder à une ressource unique (appelée section critique), mais un seul à la fois peut y accéder.

Exemple: Le carrefour

Deux véhicules empruntent régulièrement un carrefour. S'ils passent simplement sans communication (visuelle par exemple), le risque d'accident est élevé.

Ici la section critique correspond au carrefour. En effet, seul un véhicule à la fois peut y accéder, autrement c'est le crash.



printemps des sciences
Science et ville 14-20 mars 2005

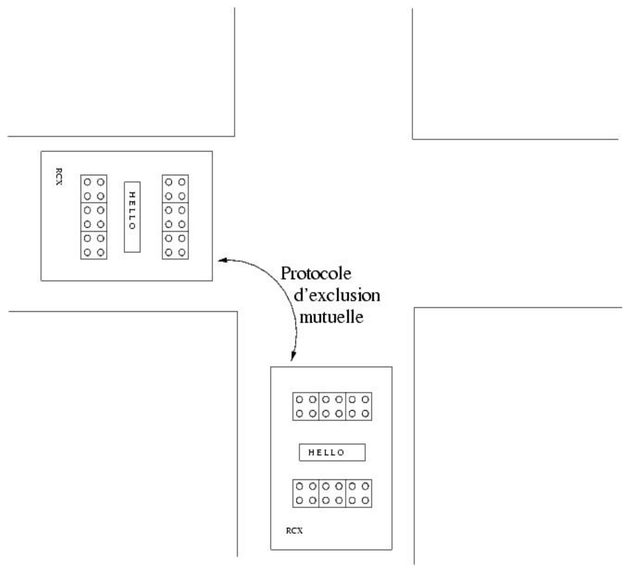
ULB

Vérification de systèmes distribués

DÉPARTEMENT D'INFORMATIQUE
David Czerwonogora – Henry Ledoux – Olivier Godart

Protocole d'exclusion mutuelle

Nos deux véhicules devront utiliser un protocole de communication, méthode de dialogue commune aux deux véhicules, qui permet à chacun de savoir s'il peut entrer sur le carrefour ou pas.



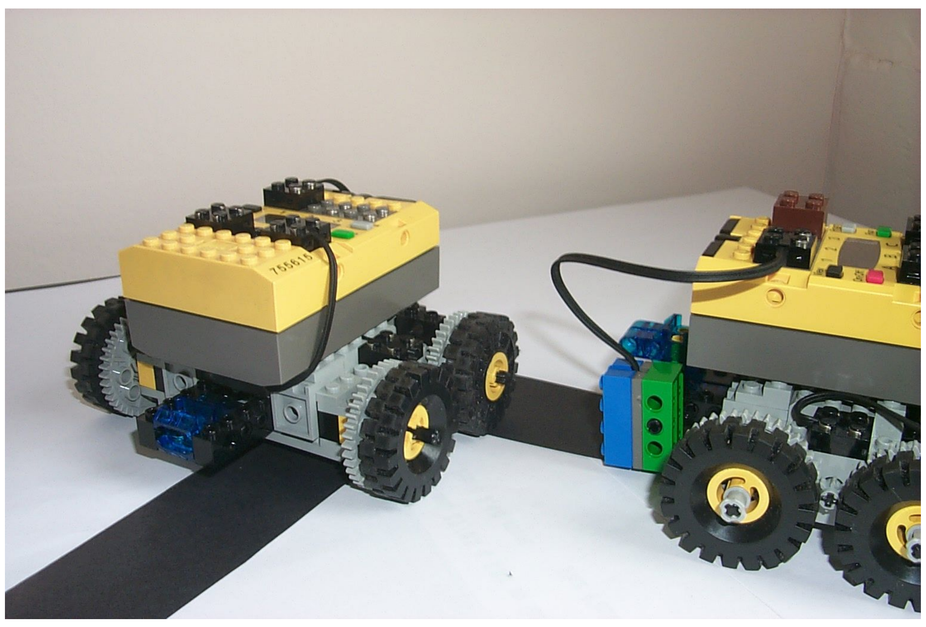
Choix du protocole

Notre protocole devra satisfaire principalement 3 propriétés :

- ✔Exclusion mutuelle : la plus évidente, et la plus importante, cette propriété spécifie qu'à tout moment : "il y a au plus un véhicule dans le carrefour".
- ✔Absence de bloquage (deadlock) : il est impossible d'arriver à un état où le carrefour est définitivement inutilisé. Cette propriété ne concerne pas directement la sécurité, mais ne pas la respecter peut avoir des conséquences coûteuses.
- ✔Absence de famine : dans le cas où un des véhicules utilise continuellement le carrefour, il ne peut en empêcher définitivement l'usage pour l'autre véhicule.

Ces propriétés s'expriment comme suit en Logique Temporelle Linéaire (LTL) :

- **[](nc <= 1)**, où nc est le nombre de véhicules sur le carrefour. Cela se lit « A tout moment, nc est inférieur ou égal à 1 ».
- **[]**◊ (nc = 1). Ceci se lit : « A tout moment, il y aura finalement un véhicule dans le carrefour ».
- **[]**(**[]**A et B →◊C) où A signifie « le véhicule 1 veut entrer en section critique », B : «2 veut entrer en section critique». et C : « le véhicule 2 est en section critique ». Autrement dit : « A tout moment : si 1 veut toujours entrer dans le carrefour et 2 aussi, alors 2 entrera finalement dans le carrefour ».



Réalisation

Nous avons réalisé une maquette illustrant le problème à l'aide de deux véhicules en Lego Mindstorm. Chaque véhicule est commandé par une brique lego (appelée RCX), contenant un microcontrôleur, que nous avons programmé. Les deux RCX suivent leur ligne noire et communiquent par infrarouge, afin de déterminer s'ils peuvent traverser le carrefour, qui correspond à la section critique.
Le protocole d'exclusion mutuelle utilisé par ces véhicules a été **vérifié ;** donc ils n'entreront **jamais** en collision dans le carrefour.

FACULTE DES SCIENCES
UNIVERSITE LIBRE DE BRUXELLES