Université Bordeaux I
U.F.R. Mathématiques et Informatique

Master Informatique 2^{eme} année
Année 2003-2004

Lundi 26 janvier 2004, 8 heures - 9 heures 30
Tous documents autorisés.

Conceptions Formelles (AltaRica)
Alain Griffault

Nom : Prénom :

Le système à contrôler

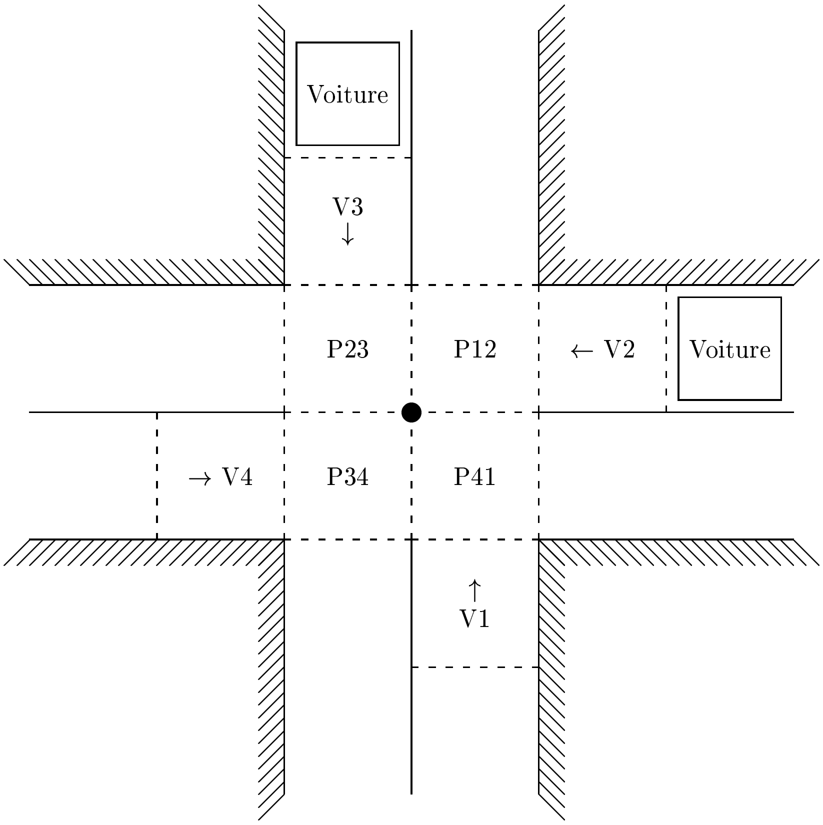

FIG. 1 – Un carrefour routier

Le système à concevoir est le contrôle de la circulation dans un carrefour tel que celui de la figure 1.Les voitures y pénètrent pour soit tourner à droite, soit continuer tout droit, soit tourner à gauche. Pour simplifier le problème, les hypothèses suivantes seront considérées valides :

- Deux véhicules ne peuvent pas être en même temps sur une des huit places *en pointillée.*
- Les routes en sortie de carrefour sont toujours libres.
- Les véhicules avancent par *bonds.* Lorsqu'un véhicule est sur une place, si la place voisine convoitée est libre, alors à l'instant suivant, si le véhicule a bougé, alors la place initiale est libre et la place convoitée est occupée.
- Les véhicules n'ont pas de marche arrière.
- En provenance de la voie $V(i)$, un véhicule doit :
  - pour tourner à droite, obtenir la place $P(i-1,i)$,
  - pour continuer tout droit, obtenir successivement la place $P(i-1,i)$ puis la place $P(i,i+1)$,
  - pour tourner à gauche, obtenir successivement la place $P(i-1,i)$ puis la place $P(i,i+1)$ puis la place $P(i+1,i+2)$.

Les propriétés attendues du système

Le contrôle devra vérifier les propriétés suivantes :

Propriété 1 *Les véhicules ne doivent jamais se gêner mutuellement de telle sorte que plus aucune d'entre elles ne puisse avancer.*

Propriété 2 *Pour des raisons de sûreté, il est interdit aux véhicules d'effectuer une marche arrière.*

Propriété 3 *Pour des raisons de sûreté, le carrefour ne doit pas être considéré comme un rond-point. Il est donc interdit aux véhicules d'utiliser le carrefour pour faire demi-tour.*

Travail à réaliser

Les six questions sont réparties dans le document joint. Vous devez répondre sur le document.

– La première question porte sur la compréhension du modèle initial.
– La deuxième question consiste à écrire les spécifications des propriétés 2 et 3.
– Les troisième, quatrième et cinquième questions portent sur la conception de modèle AltaRica.
– La sixième question porte sur la compréhension de votre modèle.

Chapitre 1

Les composants de base

1.1 Le nœud Voie

Un composant `Voie` permet de modéliser la présence d'un véhicule dans la voie, puis son entrée dans le carrefour avec l'indication de la direction souhaitée.

1.1.1 Le source AltaRica

```
node Voie
  state
    libre : bool : public;
  event
    entrer, droite, gauche, enFace;
  trans
    libre  |- entrer -> libre := false;
    ~libre |- droite, gauche, enFace -> libre := true;
  init
    libre := true;
edon
```

1.1.2 La sémantique

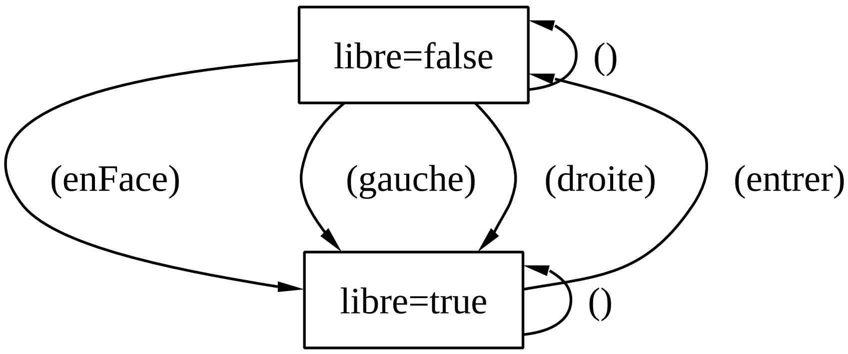

Fig. 1.1 – Le nœud Voie

1.2 Le nœud Place

Un composant `Place` modélise la présence d'un véhicule dans le carrefour et mémorise sa direction souhaitée. La variable de flux `libre` permet d'indiquer à l'environnement si la place est occupée ou non.

1.2.1 Le source AltaRica

```
node Place
  flow
    libre : bool;
  state
    droite, gauche, enFace : bool : public;
  event
    inDroite, inGauche, inEnFace,
    outDroite, outGauche, outEnFace;
  trans
    libre  |- inDroite -> droite := true;
    libre  |- inGauche -> gauche := true;
    libre  |- inEnFace -> enFace := true;
    droite |- outDroite -> droite := false;
    gauche |- outGauche -> gauche := false;
    enFace |- outEnFace -> enFace := false;
  assert
    libre = ~(droite | gauche | enFace);
  init
    droite := false, gauche := false, enFace := false;
edon
```

1.2.2 La sémantique

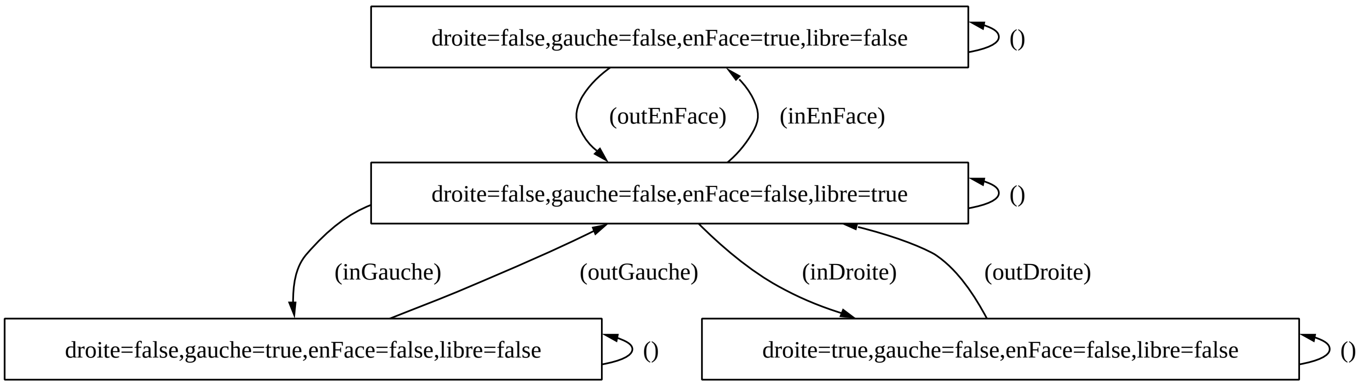

Fig. 1.2 – Le nœud Place

Chapitre 2

Un premier carrefour non contrôlé

Un système est composé :
- de quatre places indiquant la présence de voitures dans le carrefour et leur direction,
- de quatre voies modélisant les quatre files d'attente.

2.1 Le nœud SystemV0

2.1.1 Le source AltaRica

```
node SystemV0
  sub
    V1, V2, V3, V4 : Voie;
    P41, P12, P23, P34 : Place;
  sync
    // les voitures entrent dans le carrefour
    <V1.droite, P41.inDroite>;
    <V1.gauche, P41.inGauche>;
    <V1.enFace, P41.inEnFace>;
    <V2.droite, P12.inDroite>;
    <V2.gauche, P12.inGauche>;
    <V2.enFace, P12.inEnFace>;
    <V3.droite, P23.inDroite>;
    <V3.gauche, P23.inGauche>;
    <V3.enFace, P23.inEnFace>;
    <V4.droite, P34.inDroite>;
    <V4.gauche, P34.inGauche>;
    <V4.enFace, P34.inEnFace>;
    // les voitures changent de position et de direction dans le carrefour
    // pour tourner à gauche il faut ensuite aller en face
    <P41.outGauche, P12.inEnFace>;
    <P12.outGauche, P23.inEnFace>;
    <P23.outGauche, P34.inEnFace>;
    <P34.outGauche, P41.inEnFace>;
    // pour aller en face, il faut tourner à droite !
    <P41.outEnFace, P12.inDroite>;
    <P12.outEnFace, P23.inDroite>;
    <P23.outEnFace, P34.inDroite>;
    <P34.outEnFace, P41.inDroite>;
edon
```

Question 1 (1 point) *Expliquez le blocage de la figure 2.1.*

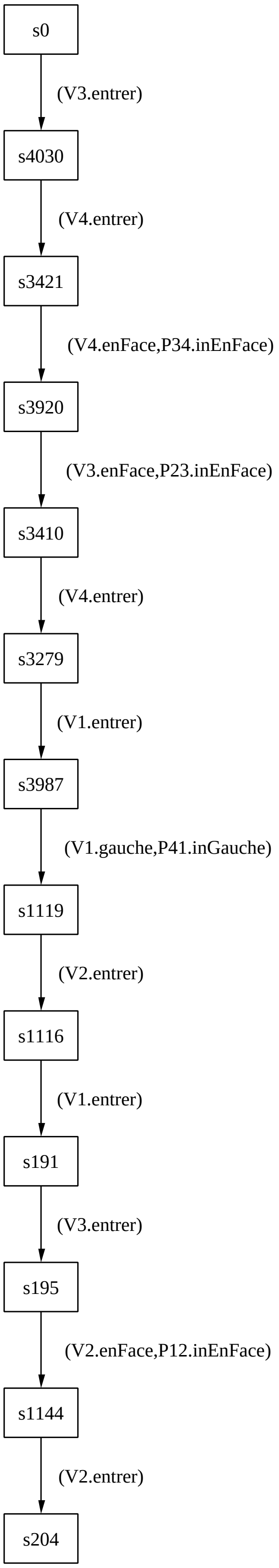

FIG. 2.1 – Une situation d'inter-blocage

2.1.2 Les propriétés

Les spécifications

Question 2 (3 points) *Complétez les spécifications* ACHECK *(*`marcheArriere` *et* `demiTour`*) du composant* `SystemV0`*, afin d'obtenir les résultats suivants.*

```
TEST(any_s=0) [FAILED] actual size = 4096
TEST(any_t=0) [FAILED] actual size = 24576
TEST(notCFC=0) [FAILED] actual size = 1024
TEST(dead=0) [FAILED] actual size = 16
TEST(deadTime=0) [FAILED] actual size = 16
TEST(unavDeadTime=0) [FAILED] actual size = 256
TEST(sControlable=0) [FAILED] actual size = 3840
TEST(tControlable=0) [FAILED] actual size = 23296
TEST(marcheArriere=0) [PASSED]
TEST(demiTour=0) [PASSED]

with SystemV0 do
 //quot()                   >  '$NODENAME.dot';
 dead                      := any_s - src(any_t - self_epsilon);
 notCFC                    := any_t - loop(any_t,any_t);
 traceDead                 := trace(initial,any_t, dead);
 dot(src(traceDead)|tgt(traceDead), traceDead) > '$NODENAME.Dead.dot';

 // des calculs auxiliaires (P41,P12,P23,P34) avec o:occupe, l:libre, x:quelconque
 olxx := [~P41.libre] & [P12.libre];
 loxx := [P41.libre] & [~P12.libre];
 xolx := [~P12.libre] & [P23.libre];
 xlox := [P12.libre] & [~P23.libre];
 xxol := [~P23.libre] & [P34.libre];
 xxlo := [P23.libre] & [~P34.libre];
 lxxo := [~P34.libre] & [P41.libre];
 oxxl := [P34.libre] & [~P41.libre];

 // une voiture va de P41 à P12
 P41toP12 := rsrc(olxx) & rtgt(loxx);

 // la propriete 2 : les voitures en marche arriere !
 marcheArriere :=









 // la propriete 3 : exemple du demi-tour P41,P12,P23,P34
 demiTour :=










 show(all)                 >  '$NODENAME.prop';
```

```
 test(dead,0)              >  '$NODENAME.res';
 test(notCFC,0)            >> '$NODENAME.res';
 test(marcheArriere,0)     >> '$NODENAME.res';
 test(demiTour,0)          >> '$NODENAME.res';
done
```

Chapitre 3

Un carrefour contrôlé par des feux

Un carrefour contrôlé est composé :
- de quatre places indiquant la présence de voitures dans le carrefour et leur direction,
- de quatre voies modélisant les quatre files d'attente.
- de quatre feux autorisant l'entrée des véhicules dans le carrefour, de telle sorte que les propriétés 1, 2 et 3 soient satisfaites.

3.1 Le nœud Feu

3.1.1 La sémantique

Question 3 (1 point) *Dessinez l'automate de la sémantique souhaitée du composant* `Feu`

3.1.2 Le source AltaRica

Question 4 (1 point) *Donnez le source* ALTARICA *du composant* `Feu`

3.2 Le nœud SystemV1

3.2.1 Le source AltaRica

Question 5 (3 points) *Complétez le source* ALTARICA *du composant* `SystemV1`

```
node SystemV1
  sub
    F1, F2, F3, F4 : Feu;
    V1, V2, V3, V4 : Voie;
    P41, P12, P23, P34 : Place;
  flow


  state


  event
    v1Entrer, v2Entrer, v3Entrer, v4Entrer;





  trans















  sync
    // Evolution des feux














    // les voitures entrent dans le carrefour
    <v1Entrer, V1.droite, P41.inDroite>;
    <v1Entrer, V1.gauche, P41.inGauche>;
    <v1Entrer, V1.enFace, P41.inEnFace>;
    <v2Entrer, V2.droite, P12.inDroite>;
```

```
    <v2Entrer, V2.gauche, P12.inGauche>;
    <v2Entrer, V2.enFace, P12.inEnFace>;
    <v3Entrer, V3.droite, P23.inDroite>;
    <v3Entrer, V3.gauche, P23.inGauche>;
    <v3Entrer, V3.enFace, P23.inEnFace>;
    <v4Entrer, V4.droite, P34.inDroite>;
    <v4Entrer, V4.gauche, P34.inGauche>;
    <v4Entrer, V4.enFace, P34.inEnFace>;
    // les voitures changent de position et de direction dans le carrefour
    // pour tourner à gauche il faut ensuite aller en face
    <P41.outGauche, P12.inEnFace>;
    <P12.outGauche, P23.inEnFace>;
    <P23.outGauche, P34.inEnFace>;
    <P34.outGauche, P41.inEnFace>;
    // pour aller en face, il faut tourner à droite !
    <P41.outEnFace, P12.inDroite>;
    <P12.outEnFace, P23.inDroite>;
    <P23.outEnFace, P34.inDroite>;
    <P34.outEnFace, P41.inDroite>;
  assert



  init





edon
```

3.2.2 Les propriétés

Question 6 (1 point) *En fonction de la confiance que vous avez dans votre conception ; indiquez (en entourant la bonne réponse) si votre composant* `SystemV1` *satisfait les propriétés du cahier des charges.*

```
TEST(dead=0)            [FAILED]  [PASSED]
TEST(notCFC=0)          [FAILED]  [PASSED]
TEST(marcheArriere=0)   [FAILED]  [PASSED]
TEST(demiTour=0)        [FAILED]  [PASSED]
```