Master 1, Conceptions Formelles
Projet du module AltaRica
Synthèse (assistée) d'un contrôleur du niveau d'une cuve

Jolliet Louis Sicardon Louis Vigneau Paul

Chapitre 1

Le sujet

1.1 Cahier des charges

Le système que l'on souhaite concevoir est composé :
— d'un réservoir contenant **toujours** suffisamment d'eau pour alimenter l'exploitation,
— d'une cuve,
— de deux canalisations parfaites amont reliant le réservoir à la cuve, et permettant d'amener l'eau à la cuve,
— d'une canalisation parfaite aval permettant de vider l'eau de la cuve,
— chaque canalisation est équipée d'une vanne commandable, afin de réguler l'alimentation et la vidange de la cuve,
— d'un contrôleur.

1.1.1 Détails techniques

La vanne

Les vannes sont toutes de même type, elles possèdent trois niveaux de débits correspondant à trois diamètres d'ouverture : 0 correspond à la vanne fermée, 1 au diamètre intermédiaire et 2 à la vanne complètement ouverte. Les vannes sont commandables par les deux instructions `inc` et `dec` qui respectivement augmente et diminue l'ouverture. Malheureusement, la vanne est sujet à défaillance sur sollicitation, auquel cas le système de commande devient inopérant, la vanne est désormais pour toujours avec la même ouverture.

La Cuve

Elle est munie de $nbSensors$ capteurs (au moins quatre) situés à $nbSensors$ hauteurs qui permettent de délimiter $nbSensors + 1$ zones. La zone 0 est comprise entre le niveau 0 et le niveau du capteur le plus bas ; la zone 1 est comprise entre ce premier capteur et le second, et ainsi de suite.

Elle possède en amont un orifice pour la remplir limité à un débit de 4, et en aval un orifice pour la vider limité à un débit de 2.

Le contrôleur

Il commande les vannes avec les objectifs suivants ordonnés par importance :

1. Le système ne doit pas se bloquer, et le niveau de la cuve ne doit jamais atteindre les zones 0 ou $nbSensors$.
2. Le débit de la vanne aval doit être le plus important possible.

On fera également l'hypothèse que les commandes ne prennent pas de temps, et qu'entre deux pannes et/ou cycle *temporel*, le contrôleur à toujours le temps de donner au moins un ordre. Réciproquement, on fera l'hypothèse que le système à toujours le temps de réagir entre deux commandes.

Les débits

Les règles suivantes résument l'évolution du niveau de l'eau dans la cuve :
— Si ($amont > aval$) alors au temps suivant, le niveau aura augmenté d'une unité.
— Si ($amont < aval$) alors au temps suivant, le niveau aura baissé d'une unité.
— Si ($amont = aval = 0$) alors au temps suivant, le niveau n'aura pas changé.
— Si ($amont = aval > 0$) alors au temps suivant, le niveau pourra :
  — avoir augmenté d'une unité,
  — avoir baissé d'une unité,
  — être resté le même.

1.2 L'étude

1.2.1 Rappel méthodologique

Comme indiqué en cours, le calcul par point fixe du contrôleur est exact, mais l'opération de projection effectuée ensuite peut perdre de l'information et générer un contrôleur qui n'est pas satisfaisant. Plus précisemment, le contrôleur AltaRica généré :
— ne garanti pas la non accessibilité des *Situations Redoutées*.
— ne garanti pas l'absence de *nouvelles situations de blocages*.
Dans le cas ou il existe toujours *des situations de blocages ou redoutées*, vous pouvez au choix :

1. Corriger manuellement le contrôleur calculé (sans doute très difficile).
2. Itérer le processus du calcul du contrôleur jusqu'à stabilisation du résultat obtenu.
   — Si le contrôleur obtenu est sans blocage et sans situation redoutée, il est alors correct.
   — Si le contrôleur obtenu contient toujours des blocages ou des situations redoutées, c'est que le contrôleur initial n'est pas assez performant, mais rien ne garanti que l'on soit capable de fournir ce premier contrôleur suffisemment performant.

Remarque : Pour vos calculs, vous pouvez utiliser au choix les commandes :
— `altarica-studio xxx.alt xxx.spe`
— `arc -b xxx.alt xxx.spe`
— `make` pour utiliser le fichier GNUmakefile fourni.

1.2.2 Le travail a réaliser

Avant de calculer les contrôleurs, vous devez répondre aux questions suivantes.

1. Expliquez le rôle de la constante $nbFailures$ et de la contrainte, présente dans le composant `System`, $nbFailures >= (V[0].fail + V[1].fail + V[2].fail)$.
2. Expliquez le rôle du composant `ValveVirtual` et de son utilisation dans le composant `CtrlVV`, afin de remplacer le composant `Ctrl` utilisé initialement.

L'étude consiste à étudier le système suivant deux paramètres :

1. $nbFailures$: une constante qui est une borne pour le nombre de vannes pouvant tomber en panne.
2. Le contrôleur initial qui peut être soit `Ctrl`, soit `CtrlVV`.

Pour chacun des huit systèmes étudiés, vous devez décrire votre méthodologie pour calculer les différents contrôleurs et répondre aux questions suivantes :

1. Est-il possible de contrôler en évitant les blocages et les situations critiques ?
2. Si oui, donnez quelques caractéristiques de ce contrôleur, si non, expliquez pourquoi.
3. Est-il possible de contrôler en optimisant le débit aval et en évitant les blocages et les situations critiques ?
4. Si oui, donnez quelques caractéristiques de ce contrôleur, si non, expliquez pourquoi.

Chapitre 2

Le rapport

Le rapport est sur 20 points.

2.1 Rôle du fichier `GNUmakefile` (2 points)

Le makefile effectue les calcules suivants :
Pour tous les contrôleurs (`Ctrl`, `CtrlVV`) et pour toutes les pannes (0,1,2,3), il concaténe tous les fichiers *alt* dans un seul et même fichier *tank.alt*. Puis il inititialise les valeurs *NBPannes* et *NomControleur* dans ces fichiers et met le résultat dans un fichier *test.alt* . De la même façon, un fichier *system.spe* est créé et copié dans *test.spe* avec plusieurs variables initialisées.
Ensuite pour toutes les iterations (0,1,2,3,4,5),le contrôleur est modifié en fonction de l'itération et le calcul est lancé. Les fichiers obtenus sont comparés et le programme coupe l'itération lorsque l'évolution stagne, c'est-à-dire quand deux itérations produisent un même résultat.

2.2 Rôle de la constante `nbFailures` et de l'assertion associée (1 point)

La constante *nbFailures* est une constante correspondant au nombre de vannes pouvant tomber en panne dans le système. Une vanne en panne est une vanne qui reste pour toujours dans son état actuel. Le système en possède trois et donc *nbFailures* peut prendre les valeurs 0, 1, 2 ou 3 selon le nombre de défaillances étudiées dans le système. La contrainte $nbFailures >= (V[0].fail + V[1].fail + V[2].fail)$ présente dans le système permet de s'assurer que le nombre de défaillances dans le système actuel ne dépasse pas le nombre de défaillances autorisées. Par exemple, si on teste le système avec une seule défaillance et qu'une vanne est déjà défaillante (V[x].fail = 1), cette contrainte va empêcher les autres vannes de dysfonctionner.

2.3 Résultats avec le contrôleur initial `Ctrl`

2.3.1 Calcul d'un contrôleur

Avec 0 défaillance (0.5 point)

```
/*
 * Properties for node : System0FCtrl
 * # state properties : 7
 *
 * any_s = 247
 * deadlock = 0
 * NC = 86
 * SR = 86
```

```
 * out0 = 80
 * out1 = 83
 * out2 = 84
 *
 * # trans properties : 4
 *
 * any_t = 3472
 * dec21 = 351
 * dec10 = 342
 * CCoupGagnant = 1134
 */

/*
 * Properties for node : System0FCtrl0F1I
 * # state properties : 7
 *
 * any_s = 94
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 26
 * out1 = 34
 * out2 = 34
 *
 * # trans properties : 4
 *
 * any_t = 858
 * dec21 = 102
 * dec10 = 68
 * CCoupGagnant = 712
 */

/*
 * Properties for node : System0FCtrl0F2I
 * # state properties : 7
 *
 * any_s = 94
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 26
 * out1 = 34
 * out2 = 34
 *
 * # trans properties : 4
 *
 * any_t = 858
 * dec21 = 102
 * dec10 = 68
 * CCoupGagnant = 712
 */
```

Interprétation des résultats Au départ on peut voir que la valve est fermée dans 80 états (out0 = 80) soit dans environ un tier d'entre eux ce qui est beaucoup car on cherche à limiter la fermeture de la valve avale et le système ne prend en compte aucune défaillance. De plus, on peut voir qu'il y a 86 niveaux critiques et 86 situations redoutées, et environ un tier des transitions correspondent à un coup gagnant (CCoupGagnant = 1134 pour any_t = 3472).

Après une itération, le contrôleur est plus performant. On peut voir que la valve se retrouve fermée moins souvent (out0 = 26). De plus, il n'y a aucun blocage ni aucune situation redoutée ou critique. Le taux de coups gagnants est également bien meilleur. Apres une seconde itération, le résultat obtenu est identique au résultat précédent et l'algorithme s'arrête. Il en sera de même pour l'ensemble des résultats suivants.

Avec 1 défaillance (0.5 point)

```
/*
 * Properties for node : System1FCtrl
 * # state properties : 7
 *
 * any_s = 958
 * deadlock = 0
 * NC = 329
 * SR = 329
 * out0 = 300
 * out1 = 326
 * out2 = 332
 *
 * # trans properties : 4
 *
 * any_t = 19540
 * dec21 = 2205
 * dec10 = 2139
 * CCoupGagnant = 4950
 */

/*
 * Properties for node : System1FCtrl1F1I
 * # state properties : 7
 *
 * any_s = 508
 * deadlock = 93
 * NC = 69
 * SR = 93
 * out0 = 120
 * out1 = 188
 * out2 = 200
 *
 * # trans properties : 4
 *
 * any_t = 5230
 * dec21 = 695
 * dec10 = 443
 * CCoupGagnant = 2941
 */

/*
 * Properties for node : System1FCtrl1F2I
 * # state properties : 7
 *
 * any_s = 508
 * deadlock = 96
 * NC = 69
 * SR = 96
 * out0 = 120
```

```
 * out1 = 188
 * out2 = 200
 *
 * # trans properties : 4
 *
 * any_t = 5161
 * dec21 = 695
 * dec10 = 439
 * CCoupGagnant = 2909
 */

/*
 * Properties for node : System1FCtrl1F3I
 * # state properties : 7
 *
 * any_s = 508
 * deadlock = 96
 * NC = 69
 * SR = 96
 * out0 = 120
 * out1 = 188
 * out2 = 200
 *
 * # trans properties : 4
 *
 * any_t = 5161
 * dec21 = 695
 * dec10 = 439
 * CCoupGagnant = 2909
 */
```

Interprétation des résultats Suite à l'ajout d'une défaillance, on remarque tout d'abord que le nombre d'états et de transitions augmente grandement ainsi que le nombre de niveaux critiques (NC = 329) et situations redoutées (SR = 329). Malgré tout, la proportion d'états où la valve avale est fermée est approximativement la même que lorsqu'il n'y a pas de défaillance (out0 = 300 ce qui correspond environ un tier des états). En revanche, le nombre de coups gagnants par rapport au nombre de transitions est faible (CCoupGagnant = 4950 pour any_t = 19450, soit environ un quart seulement) et un grand nombre d'états mènent à une situation où la valve avale est fermée.
Après une itération, le nombre de niveaux critiques (NC = 69) et de situations redoutées (SR = 93) diminue grandement. De plus, plus de la moitié des transitions sont des coups gagants (any_t = 2941 pour any_t = 5230). En revanche, on voit apparaître des situations bloquantes (deadlock = 93). On s'aperçoit ici que le nombre de situations redoutées correspond au nombre de situations bloquantes (SR = deadlock). Il y a également moins d'états dans lesquels la valve avale est fermée que précédement en terme de proportion (le nombre total d'états diminue donc il est normal que le nombre d'états out0 diminue).
Après une deuxième itération, les seuls changements que l'on peut remarquer sont au niveau des situations bloquantes et des transitions. Cette itération a diminué le nombre de transitions total, en enlevant 40 transitions (dec10 passe de 5285 à 5245) et 32 coups gagnants (CCoupGagnant passe de 2941 à 2909). Cela a eu pour effet d'augmenter les situations bloquantes (deadlock passe de 93 à 96) et donc de situations redoutées (SR passe de 93 à 96).

Avec 2 défaillances (0.5 point)

```
/*
 * Properties for node : System2FCtrl
 * # state properties : 7
 *
```

```
 * any_s = 1627
 * deadlock = 0
 * NC = 551
 * SR = 551
 * out0 = 506
 * out1 = 553
 * out2 = 568
 *
 * # trans properties : 4
 *
 * any_t = 44608
 * dec21 = 5418
 * dec10 = 5228
 * CCoupGagnant = 7533
 */

/*
 * Properties for node : System2FCtrl2F1I
 * # state properties : 7
 *
 * any_s = 790
 * deadlock = 239
 * NC = 107
 * SR = 239
 * out0 = 200
 * out1 = 306
 * out2 = 284
 *
 * # trans properties : 4
 *
 * any_t = 7168
 * dec21 = 725
 * dec10 = 834
 * CCoupGagnant = 3029
 */

/*
 * Properties for node : System2FCtrl2F2I
 * # state properties : 7
 *
 * any_s = 774
 * deadlock = 271
 * NC = 107
 * SR = 271
 * out0 = 190
 * out1 = 302
 * out2 = 282
 *
 * # trans properties : 4
 *
 * any_t = 6547
 * dec21 = 578
 * dec10 = 824
 * CCoupGagnant = 2826
 */
```

```
/*
 * Properties for node : System2FCtrl2F3I
 * # state properties : 7
 *
 * any_s = 772
 * deadlock = 270
 * NC = 107
 * SR = 270
 * out0 = 190
 * out1 = 302
 * out2 = 280
 *
 * # trans properties : 4
 *
 * any_t = 6534
 * dec21 = 578
 * dec10 = 824
 * CCoupGagnant = 2826
 */

/*
 * Properties for node : System2FCtrl2F4I
 * # state properties : 7
 *
 * any_s = 772
 * deadlock = 270
 * NC = 107
 * SR = 270
 * out0 = 190
 * out1 = 302
 * out2 = 280
 *
 * # trans properties : 4
 *
 * any_t = 6534
 * dec21 = 578
 * dec10 = 824
 * CCoupGagnant = 2826
 */
```

Interprétation des résultats Avec une deuxième défaillance, on constate encore une fois une augmentation du nombre d'états (any_s = 1627) ce qui a pour conséquence de faire augmenter l'ensemble des autres valeurs. Les proportions restent à peu près similaires aux résultats précédemment (première itération avec 0 et 1 défaillance) avec un nombre de situations redoutées équivalent au nombre de niveaux critiques et des états *out0*, *out1* et *out2* en quantité presque similaire (un tier de chaque). En revanche, cette fois le nombre de transitions any_t est supérieur aux valeurs de dec21 et dec10 et la proportion de coups gagnants est plus faible (environ un sixième des transitions).
Après la première itération on remarque qu'il y a un grand nombre de situations bloquantes (deadlock = 239) et ici aussi ce nombre correspond à celui des situations redoutées (deadlock = SR). On à également le nombre de coup gagnants *CCoupGagnant* qui augmente par rapport au nombre de transitions *any_t*. La deuxième itération réduit le nombre d'états notamment ceux ou la valve est fermée (*out0*) et le nombre de décrémentation du débit de la valve du niveau 2 à 1(*dec21*). Au final après une troisième itération, on supprime deux états où la valve à un débit de 2 ce qui réduit de 1 le nombre de situations bloquantes.

Avec 3 défaillances (0.5 point)

```
/*
 * Properties for node : System3FCtrl
 * # state properties : 7
 *
 * any_s = 1832
 * deadlock = 0
 * NC = 617
 * SR = 617
 * out0 = 570
 * out1 = 622
 * out2 = 640
 *
 * # trans properties : 4
 *
 * any_t = 57696
 * dec21 = 7242
 * dec10 = 6974
 * CCoupGagnant = 7908
 */

/*
 * Properties for node : System3FCtrl3F1I
 * # state properties : 7
 *
 * any_s = 240
 * deadlock = 112
 * NC = 0
 * SR = 112
 * out0 = 48
 * out1 = 120
 * out2 = 72
 *
 * # trans properties : 4
 *
 * any_t = 1568
 * dec21 = 84
 * dec10 = 96
 * CCoupGagnant = 343
 */

/*
 * Properties for node : System3FCtrl3F2I
 * # state properties : 7
 *
 * any_s = 62
 * deadlock = 27
 * NC = 0
 * SR = 27
 * out0 = 36
 * out1 = 26
 * out2 = 0
 *
 * # trans properties : 4
 *
 * any_t = 609
 * dec21 = 0
```

```
 * dec10 = 0
 * CCoupGagnant = 343
 */

/*
 * Properties for node : System3FCtrl3F3I
 * # state properties : 7
 *
 * any_s = 62
 * deadlock = 27
 * NC = 0
 * SR = 27
 * out0 = 36
 * out1 = 26
 * out2 = 0
 *
 * # trans properties : 4
 *
 * any_t = 609
 * dec21 = 0
 * dec10 = 0
 * CCoupGagnant = 343
 */
```

Interprétation des résultats

2.3.2 Calcul des contrôleurs optimisés (2 points)

```
event
  /*
   * les priorites dependent des actions sur la vanne aval
   * inc > nop > dec
   */
  {ddi, dii, dni, idi, iii, ini, ndi, nii, nni} >
    {ddn, din, dnn, idn, iin, inn, ndn, nin, nnn};
  {ddn, din, dnn, idn, iin, inn, ndn, nin, nnn} >
    {ddd, did, dnd, idd, iid, ind, ndd, nid, nnd};
edon
```

Dans la description du contrôleur en 1.1.3 il est écrit que le débit de la vanne avale doit être le plus important possible. Le fichier optimisation.alt permet de définir des priorités sur le choix des actions. On va préférer incrémenter le débit avale plupart que le laisser au même niveau et on va préférer le laisser au même niveau plutôt que le décrémenter. De ce fait, on aura toujours le débit avale qui sera le plus élevé possible.

Avec 0 défaillance

```
/*
 * Properties for node : System0FCtrl0F2I_Opt
 * # state properties : 7
 *
 * any_s = 49
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 1
 * out1 = 14
 * out2 = 34
```

```
 *
 * # trans properties : 4
 *
 * any_t = 220
 * dec21 = 1
 * dec10 = 0
 * CCoupGagnant = 142
 */
```

Sans aucune défaillance, il n'y a pas de cas critique, et on observe qu'effectivement, qu'il y a beaucoup plus de somment ayant une valve avale soit ouverte (34), soit semi-ouverte (14). Il ne reste qu'un seul état ayant une valve fermée. On observe également qu'il n'y a aucune transition allant d'une valve semi-ouverte vers une valve fermée. De la même façon on a également un grand nombre de coup gagnant (142 sur 220 possibles). L'optimisation fonctionne donc très bien et nous propose un contrôleur bien plus efficace que le précédent.

2.4 Construction d'un contrôleur initial plus performant

2.4.1 Rôle du composant `ValveVirtual`(2 points)

Le composant `ValveVirtual` correspond à une valve parfaite. Il est possible d'incrémenter ou de décrémenter l'ouverture de la valve afin de laisser passer plus ou moins d'eau. La valve se stoppe (reste dans son état) dans le cas où il y aurait une défaillance.

Contrairement au composant `Valve`, il ne gère pas lui-même s'il y a une défaillance. Il compare les valeurs *rate* et *rateReal*, cette dernière valeur étant donnée par `CtrlVV`. Dans le cas où ces deux dernières valeurs sont différentes, la valve reste inactive, mais n'a pas de variable propre au caractère de blocage comme dans `Valve`.

2.4.2 Rôle du composant `CtrlVV` (5 points)

Le `CtrlVV`, contrairement au `Ctrl` précédent, contient 3 `ValveVirtual`. Il contient aussi 3 *rate* compris entre 0 et 2 correspondant aux 3 valves ainsi que le niveau actuel de la cuve entre O et *nbSensors*. A tout instant, le contrôleur teste si le *rateReal* connu par chaque valve correspond au *rate* du contrôleur. Si ce n'est pas le cas, le contrôleur détecte une annomalie avec un ASSERT.

Ensuite, il définit la liste des 27 événements possibles pour `CtrlVV` (ddd, ddi, ddn...) qui correspondent à toutes les combinaisons des actions sur les 3 valves.

A chacun de ces événements, une suite d'opération est exécutée. Par exemple, pour *ddd* le contrôleur décrémente chacune des valves une à une.

Le `CtrlVV` présente comme avantage qu'il gère lui-même les `ValveVirtual` et donc que le traitement amènera à plus de performances.

2.5 Résultats avec le contrôleur `CtrlVV`

2.5.1 Calcul d'un contrôleur

Avec 0 défaillance (0.5 point)

```
/*
 * Properties for node : System0FCtrlVV
 * # state properties : 7
 *
 * any_s = 247
 * deadlock = 0
 * NC = 86
 * SR = 86
 * out0 = 80
```

```
 * out1 = 83
 * out2 = 84
 *
 * # trans properties : 4
 *
 * any_t = 1863
 * dec21 = 213
 * dec10 = 208
 * CCoupGagnant = 548
 */

/*
 * Properties for node : System0FCtrlVV0F1I
 * # state properties : 7
 *
 * any_s = 94
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 26
 * out1 = 34
 * out2 = 34
 *
 * # trans properties : 4
 *
 * any_t = 508
 * dec21 = 65
 * dec10 = 44
 * CCoupGagnant = 362
 */

/*
 * Properties for node : System0FCtrlVV0F2I
 * # state properties : 7
 *
 * any_s = 94
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 26
 * out1 = 34
 * out2 = 34
 *
 * # trans properties : 4
 *
 * any_t = 508
 * dec21 = 65
 * dec10 = 44
 * CCoupGagnant = 362
 */
```

Interprétation des résultats Avec ce nouveau contrôleur muni de valves "virtuelles", et avec aucune défaillance, on obtient les mêmes propriétés d'état qu'avec le contrôleur `Ctrl`. Il n'y a que les propriétés de transitions qui changent, globalement elles sont toutes divisées par 2. On obtient cependant un ratio de coups gagnant / transitions possibles légèrement plus élevé.
Lors de la seconde occurrence et tout comme le le premier contrôleur, on obtient plus aucun Deadlock,

Situation Redoutée ou Niveau Critique. On obtient cependant, moins de transitions, moins de transitions permettant de décrémenter mais également moins de coups gagnants.

Avec 1 défaillance (0.5 point)

```
/*
 * Properties for node : System1FCtrlVV
 * # state properties : 7
 *
 * any_s = 1201
 * deadlock = 0
 * NC = 413
 * SR = 413
 * out0 = 350
 * out1 = 463
 * out2 = 388
 *
 * # trans properties : 4
 *
 * any_t = 8370
 * dec21 = 910
 * dec10 = 888
 * CCoupGagnant = 1866
 */

/*
 * Properties for node : System1FCtrlVV1F1I
 * # state properties : 7
 *
 * any_s = 316
 * deadlock = 16
 * NC = 0
 * SR = 16
 * out0 = 68
 * out1 = 138
 * out2 = 110
 *
 * # trans properties : 4
 *
 * any_t = 1076
 * dec21 = 92
 * dec10 = 64
 * CCoupGagnant = 546
 */

/*
 * Properties for node : System1FCtrlVV1F2I
 * # state properties : 7
 *
 * any_s = 232
 * deadlock = 3
 * NC = 0
 * SR = 3
 * out0 = 46
 * out1 = 104
 * out2 = 82
```

```
 *
 * # trans properties : 4
 *
 * any_t = 787
 * dec21 = 52
 * dec10 = 36
 * CCoupGagnant = 413
 */

/*
 * Properties for node : System1FCtrlVV1F3I
 * # state properties : 7
 *
 * any_s = 224
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 46
 * out1 = 104
 * out2 = 74
 *
 * # trans properties : 4
 *
 * any_t = 745
 * dec21 = 44
 * dec10 = 36
 * CCoupGagnant = 392
 */

/*
 * Properties for node : System1FCtrlVV1F4I
 * # state properties : 7
 *
 * any_s = 224
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 46
 * out1 = 104
 * out2 = 74
 *
 * # trans properties : 4
 *
 * any_t = 745
 * dec21 = 44
 * dec10 = 36
 * CCoupGagnant = 392
 */
```

Interprétation des résultats Lorsqu'il y a une défaillance, on obtient aucun deadlock, mais plusieurs Situations Redoutées et Niveaux Critiques. On a également moins de sommets contenant une vanne avale fermée, comparés aux autres niveaux d'ouverture. En ce qui concerne les transitions, on obtient en obtient nettement moins qu'avec le contrôleur `Ctrl`.
Au fur et à mesure des itérations, le nombre de deadlocks et dr SR/NC diminue, jusqu'à être nul. On obtient également un nombre d'états où la valve avale est fermée inférieur aux autres. Comme la tendance précédente, on obtient moins de transitions que dans le contrôleur précédent.

Avec 2 défaillances (0.5 point)

```
/*
 * Properties for node : System2FCtrlVV
 * # state properties : 7
 *
 * any_s = 2398
 * deadlock = 0
 * NC = 812
 * SR = 812
 * out0 = 651
 * out1 = 1005
 * out2 = 742
 *
 * # trans properties : 4
 *
 * any_t = 15894
 * dec21 = 1666
 * dec10 = 1625
 * CCoupGagnant = 2360
 */

/*
 * Properties for node : System2FCtrlVV2F1I
 * # state properties : 7
 *
 * any_s = 274
 * deadlock = 70
 * NC = 0
 * SR = 70
 * out0 = 52
 * out1 = 130
 * out2 = 92
 *
 * # trans properties : 4
 *
 * any_t = 725
 * dec21 = 30
 * dec10 = 27
 * CCoupGagnant = 155
 */

/*
 * Properties for node : System2FCtrlVV2F2I
 * # state properties : 7
 *
 * any_s = 2
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 2
 * out1 = 0
 * out2 = 0
 *
 * # trans properties : 4
 *
```

```
 * any_t = 4
 * dec21 = 0
 * dec10 = 0
 * CCoupGagnant = 1
 */

/*
 * Properties for node : System2FCtrlVV2F3I
 * # state properties : 7
 *
 * any_s = 2
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 2
 * out1 = 0
 * out2 = 0
 *
 * # trans properties : 4
 *
 * any_t = 4
 * dec21 = 0
 * dec10 = 0
 * CCoupGagnant = 1
 */
```

Interprétation des résultats En ajoutant une défaillance, on obtient beaucoup plus de Niveau Critiques et Situation Redoutées (812), ainsi que beaucoup de transitions (15894), dont très peu menant à un Coup Gagant (2360).
Cependant, après une itération, le nombre de Niveaux critiques devient nul, mais un grand nombre de deadlock apparaît (70) sur seulement 274 états. On obtient également moins de sommets contenant une valve avale fermée (52) Concernant les transitions, il ya moins de transitions (725) mais toujours très peu menant à un Coup Gagnant (155).
Pour finir, la seconde itération nous amène aucun deadlock, mais il n'y a que 2 sommets, les deux contenants une valve avale fermée.

Avec 3 défaillances (0.5 point)

```
/*
 * Properties for node : System3FCtrlVV
 * # state properties : 7
 *
 * any_s = 2889
 * deadlock = 0
 * NC = 970
 * SR = 970
 * out0 = 764
 * out1 = 1253
 * out2 = 872
 *
 * # trans properties : 4
 *
 * any_t = 18776
 * dec21 = 1938
 * dec10 = 1890
 * CCoupGagnant = 2384
 */
```

```
/*
 * Properties for node : System3FCtrlVV3F1I
 * # state properties : 7
 *
 * any_s = 210
 * deadlock = 97
 * NC = 0
 * SR = 97
 * out0 = 36
 * out1 = 114
 * out2 = 60
 *
 * # trans properties : 4
 *
 * any_t = 565
 * dec21 = 26
 * dec10 = 16
 * CCoupGagnant = 27
 */

/*
 * Properties for node : System3FCtrlVV3F2I
 * # state properties : 7
 *
 * any_s = 2
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 2
 * out1 = 0
 * out2 = 0
 *
 * # trans properties : 4
 *
 * any_t = 4
 * dec21 = 0
 * dec10 = 0
 * CCoupGagnant = 1
 */

/*
 * Properties for node : System3FCtrlVV3F3I
 * # state properties : 7
 *
 * any_s = 2
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 2
 * out1 = 0
 * out2 = 0
 *
 * # trans properties : 4
 *
 * any_t = 4
 * dec21 = 0
```

```
 * dec10 = 0
 * CCoupGagnant = 1
 */
```

Interprétation des résultats Avec 3 defaillances, le controleur a exactement le même comportement que précédemment, mais il y a besoin de moins d'itération. Au début, beaucoup de NC et SR, ainsi qu'un nombre de sommet contenant une valve avale fermée plus faible que les autres (valves). Puis après itération, on obtient exactement le même résultat que précédemment.

2.5.2 Calcul des contrôleurs optimisés (2 points)

Avec 0 défaillance

```
/*
 * Properties for node : System0FCtrlVV0F2I_Opt
 * # state properties : 7
 *
 * any_s = 49
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 1
 * out1 = 14
 * out2 = 34
 *
 * # trans properties : 4
 *
 * any_t = 174
 * dec21 = 1
 * dec10 = 0
 * CCoupGagnant = 96
 */
```

Avec 1 défaillance

```
/*
 * Properties for node : System1FCtrlVV1F4I_Opt
 * # state properties : 7
 *
 * any_s = 191
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 21
 * out1 = 96
 * out2 = 74
 *
 * # trans properties : 4
 *
 * any_t = 580
 * dec21 = 16
 * dec10 = 14
 * CCoupGagnant = 277
 */
```

Avec 2 défaillances

```
/*
 * Properties for node : System2FCtrlVV2F3I_Opt
 * # state properties : 7
 *
 * any_s = 2
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 2
 * out1 = 0
 * out2 = 0
 *
 * # trans properties : 4
 *
 * any_t = 4
 * dec21 = 0
 * dec10 = 0
 * CCoupGagnant = 1
 */
```

Avec 3 défaillances

```
/*
 * Properties for node : System3FCtrlVV3F3I_Opt
 * # state properties : 7
 *
 * any_s = 2
 * deadlock = 0
 * NC = 0
 * SR = 0
 * out0 = 2
 * out1 = 0
 * out2 = 0
 *
 * # trans properties : 4
 *
 * any_t = 4
 * dec21 = 0
 * dec10 = 0
 * CCoupGagnant = 1
 */
```

2.6 Conclusion (2 points)