Commande tolérante aux fautes des systèmes à événements discrets : comparaison de deux approches sur un cas d'étude

Julien Niguez, Saïd Amari, Jean-Marc Faure

Commande tolérante aux fautes des systèmes à événements discrets : comparaison de deux approches sur un cas d'étude

Julien Niguez[1], Saïd Amari[1], Jean-Marc Faure[1]

[1]Laboratoire Universitaire de Recherche en Production Automatisée
LURPA, ENS Cachan, Univ Paris-Sud, F-94235 Cachan, France
{julien.niguez ; said.amari ; jean-marc.faure}@ens-cachan.fr

Résumé— **Dans ce papier, deux approches de commande tolérante aux fautes pour les systèmes à événements discrets sont évaluées : l'approche par masquage de fautes [9] et l'approche par reconfiguration du contrôle [3]. Elles ont été appliquées à un cas d'étude, afin de pouvoir être comparées. Le détail des applications est présenté, ainsi qu'une discussion autour de ces applications et de la capacité de traitement des fautes des deux méthodes.**

Mots-clés— **Systèmes à événements discrets, commande tolérante aux fautes, reconfiguration de contrôle, masquage de fautes**

I. Introduction

La productivité au sein d'une entreprise est un enjeu majeur, aux implications économiques importantes. Pour maintenir une bonne productivité, il est nécessaire que les moyens de production aient une grande disponibilité. Cette dernière dépend entre autre de la capacité du système à s'adapter aux fautes avant qu'elles aient un impact négatif sur sa production. Les méthodes de Commande Tolérante aux Fautes (CTF par la suite) permettent d'agir sur le contrôleur du système, de manière à intégrer un comportement défaillant des composants du procédé. Cela permet d'adapter la stratégie de production avant réduction de la productivité du système.

Les principales techniques en diagnostic des fautes et en CTF pour les systèmes automatisés continus sont exposées dans [1]. Les nombreux aspects du CTF appliqués aux systèmes continus sont traités et illustrés à l'aide d'exemples. Dans sa deuxième version, le livre propose une méthode de CTF adapté aux Systèmes à Événements Discrets (SED) basée sur la reconfiguration du contrôle, ainsi que plusieurs exemples l'illustrant. Plus récemment, plusieurs méthodes de CTF pour les SED ont été développées, utilisant divers formalismes de représentations de ces derniers. Celles que nous avons choisi de présenter font toutes l'hypothèse de fautes non-réparables.

Une première approche de contrôle par superviseur tolérant aux fautes pour les Automates Finis (AF) est présentée dans [7]. Utilisant le concept de stabilité et de convergence des langages, l'objectif est de modéliser un superviseur capable d'assurer un comportement après faute équivalent au comportement nominal du système, en un nombre fini d'étapes. Ces comportements sont dit équivalents s'ils génèrent le même langage marqué. Une autre méthode utilisant la théorie de supervision est proposé dans [8], dont le but est d'obtenir un superviseur dit d'accommodation aux fautes. Ce superviseur est construit à partir du comportement nominal du procédé, auquel on vient rajouter le comportement attendu après la faute. De par sa construction, il permet de superviser le contrôleur après occurrence d'une faute sans utilisation d'un diagnostiqueur pour détecter cette dernière. Une méthode de synthèse de contrôleur tolérant aux fautes pour les SED modélisés par des AF est proposée dans [5]. Pour ce faire, un contrôleur-diagnostiqueur est défini, dont le but est de détecter la faute avant l'exécution de séquences illégales, forcer l'arrêt du système après le diagnostic de la faute, puis soumettre de nouvelles spécifications au système fautif. En se basant sur les Réseaux de Petri, [6] présente une méthode de synthèse de superviseur tolérant aux fautes. Les spécifications sont alors décrites à l'aide de marquages interdits. Enfin, les méthodes de masquage de fautes introduite dans [9] et de reconfiguration pour les Automates Entrée/Sortie (AES) introduite dans [3] seront présentées dans les parties III et IV.

Afin de pouvoir comparer ces approches, il est nécessaire de les appliquer à un même cas d'étude. Nous en avons donc sélectionné deux avant de les illustrer avec un système. Les méthodes diffèrent par la présence ou non d'un diagnostiqueur pour détecter l'occurrence d'une faute. Nous avons donc choisi de comparer une méthode utilisant un diagnostiqueur à une méthode n'en utilisant pas. Parmi celles répondant à ce critère, les plus récentes ont été choisies. Ces deux méthodes ont été appliquées pour le traitement de différents types de fautes. Une comparaison des deux approches autour de ces applications est ensuite proposée, en termes de capacité de traitement des fautes, de difficultés de mise en oeuvre et d'existence d'outils logiciels supportant la méthode.

La suite de ce papier s'organise de la manière suivante : la partie II présente le système sur lequel ont été illustrées les deux méthodes choisies. Les parties III et IV proposent une présentation des méthodes choisies. La partie V propose le détail de l'application des deux méthodes. Enfin, la partie VI expose une discussion autour de l'application des méthodes au système, et de leur capacité à traiter les fautes sélectionnées.

II. Présentation du cas d'étude

Le système utilisé pour la comparaison des méthodes sélectionnées est inspiré du système de tri de colis présent dans le logiciel de simulation de parties opératives ITS PLC [11]. Afin de pouvoir adapter ou reconfigurer la stratégie de contrôle en cas de défaillance d'un des composant du système, des modifications ont été apportées au système

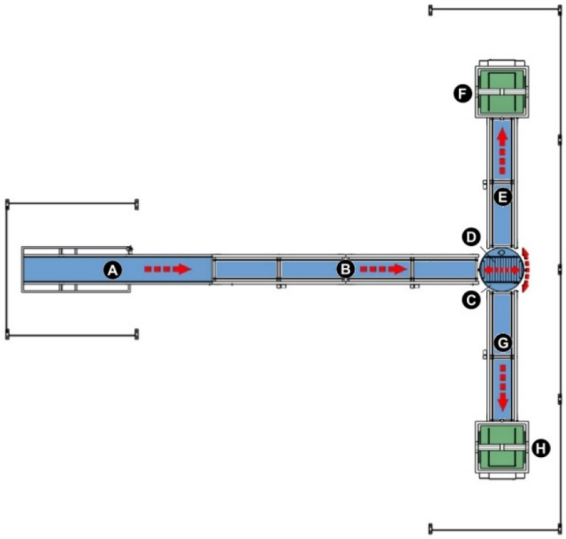

Fig. 1. Le système de tri de colis considéré

(ajout de capteurs et actionneurs). Le système est constitué d'un convoyeur d'entrée (repère A sur la figure 1), qui achemine les colis jusqu'au convoyeur intermédiaire (B). Les colis sont ensuite acheminés jusqu'à une table tournante (C), sur laquelle se trouvent des rouleaux (D). Les convoyeurs de sortie (E et G) et les ascenseurs (F et H) ne sont pas considérés dans la suite.

Des colis de deux tailles sont acheminés via le tapis d'entrée : des *petits colis* et des *grands colis*. L'objectif du système est de distribuer les grands colis à gauche et les petits colis à droite.

A. Fonctionnement du système

Pour réaliser son objectif, le système dispose des actionneurs et capteurs suivants, respectivement présentés dans les tableaux I et II.

TABLE I
ACTIONNEURS DU SYSTÈME ET ÉVÉNEMENTS ASSOCIÉS

Actionneur	Événement
Rotation du convoyeur A	A
Rotation du convoyeur B	B
Rotation des rouleaux - sens horaire	R_h
Rotation des rouleaux - sens anti-horaire	R_a
Rotation de la table - gauche	T_g
Rotation de la table - droite	T_d

Pour la suite, on associera les commandes actionneurs à des événements contrôlables et les informations capteurs à des événements incontrôlables.

Le fonctionnement du système est le suivant. Les colis arrivent de manière aléatoire sur le convoyeur A. Les colis en fin de convoyeur A sont envoyés sur le convoyeur B si celui si est libre. S'il ne l'est pas, le convoyeur A est arrêté jusqu'à libération du convoyeur B. Ce dernier est arrêté quand un colis arrive en fin de convoyeur B mais que la table n'est pas en position centrale. Un colis est chargé sur la table grâce à la rotation des rouleaux dans le sens horaire. Une fois la table chargée, elle est tournée en position

TABLE II
CAPTEURS DU SYSTÈME ET ÉVÉNEMENTS ASSOCIÉS

Capteur	Événement
Colis en fin de convoyeur A	c_a
Colis en fin de convoyeur B	c_b
Grand colis sur table	c_g
Petit colis sur table	c_p
Table en position gauche	p_g
Table en position centrale	p_m
Table en position droite	p_d
Colis déchargé à droite	d_d
Colis déchargé à gauche	d_g

gauche. Les grands colis (respectivement petits colis) sont alors déchargés à gauche (à droite) en utilisant la rotation dans le sens horaire (anti-horaire) des rouleaux. La table retourne ensuite en position centrale, prête à recevoir un nouveau colis.

B. Classification des fautes

Les fautes seront considérées comme non-réparables et modélisées par des événements non observables. On ne considère pas d'occurrences simultanées de fautes. Afin de simuler différents types de fautes, la classification suivante des fautes est proposée :

- Les *fautes actionneurs* sont des fautes qui impliquent un actionneur (vérin, moteur), un pré-actionneur (distributeur, contacteur), une connexion du pré-actionneur au contrôleur ou la carte de sortie du contrôleur.
- Les *fautes capteurs* sont des fautes qui impliquent un capteur, une connexion de capteur au contrôleur ou la carte d'entrée du contrôleur.
- Les *fautes processus* sont des fautes qui impliquent le processus et son environnement (par exemple, la chute d'un colis).

On choisit d'illustrer la faute de type *actionneur* par une défaillance du pré-actionneur contrôlant la mise en rotation dans le sens anti-horaire des rouleaux. La faute de type *capteur* est illustrée par une défaillance du capteur en fin de convoyeur A. Cependant, une faute de type *processus* ne peut toutefois pas être traitée avec la modélisation choisie. Une absence d'événement est en pratique traitée par la mise en place d'une temporisation. Les formalisme choisis étant non-temporisés, on ne peut traiter ce genre de fautes.

Dans la suite, une faute sera modélisée par l'événement f, tel que $f \in \Sigma_u$ est non-observable.

C. Stratégie de reconfiguration

De par sa construction, le système permet de réaliser son objectif dans le cas où l'une des fautes considérées ci-dessus surviendrait.

Dans le cas de la faute de type *actionneur*, le fonctionnement du système devient le suivant : un colis est chargé sur la table grâce à la rotation des rouleaux dans le sens horaire. Une fois la table chargée, elle est tournée en position gauche (respectivement droite) si le colis chargé est un grand (petit) colis. La table est ensuite déchargée en utilisant la rotation des rouleaux dans le sens horaire, puis retourne en position centrale.

Dans le cas de la faute de type *capteur*, il n'est plus possible de détecter un colis en fin du convoyeur A. Le système ne possédant pas de redondance au niveau du capteur, un comportement dégradé est spécifié de manière à éviter les collisions entre colis. Le convoyeur A est donc arrêté dès qu'un colis est détecté en fin de convoyeur B et que la table n'est pas prête, qu'il y ai un colis en fin du convoyeur A ou non.

III. Approche par masquage de Fautes

A. Principe de la méthode de masquage de fautes

La méthode de masquage de fautes introduite dans [9] vient intercaler un masqueur de fautes entre les composants du procédé et le contrôleur (Figure 2). Ce masqueur de fautes a pour objectif d'interpréter les échanges entre le contrôleur et le procédé. Si une faute intervient, le masqueur peut alors modifier les informations transmises, de manière à simuler pour le procédé un contrôleur qui prend en compte la faute.

Cette méthode possède deux caractéristiques la distinguant des autres : elle ne nécessite pas l'utilisation un diagnostiqueur afin de détecter l'occurrence de la faute, et elle ne vient pas modifier les modèles d'origine du procédé et du contrôleur.

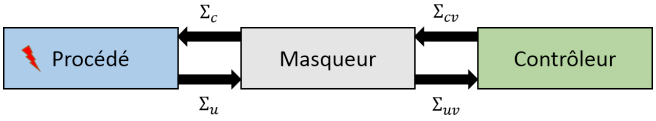

Fig. 2. CTF avec *masqueur de fautes*

Pour réaliser la commande tolérante aux fautes, on introduit un alphabet *virtuel* $\Sigma_v = \Sigma_{cv} \cup \Sigma_{uv}$, en bijection avec l'alphabet Σ. En fonction des informations qu'il reçoit en entrée (Σ_u et Σ_{cv}), le masqueur de fautes adapte les informations qu'il émet (Σ_c et Σ_{uv}).

Pour l'illustration, on ne s'intéresse donc qu'à la construction du masqueur. Le modèle du contrôleur est supposé connu. Le formalisme utilisé pour cette approche est celui des automates à états finis.

B. Automates à états finis

Un *automate à états finis*, noté G, est défini par le 5-uplet

$$G = (X, \Sigma, \delta, x_0, X_m) \quad (1)$$

avec X l'ensemble supposé fini des états, Σ l'ensemble supposé fini des événements, $\delta : X \times \Sigma \to X$ la fonction de transition $\delta(x, e) = y$ signifiant qu'il existe une transition labellisée par l'événement e allant de l'état x à l'état y, x_0 l'état initial $x_0 \in X$ et X_m l'ensemble d'états marqués $X_m \subset X$.

L'ensemble des événements Σ peut être décomposé de la manière suivante $\Sigma = \Sigma_c \cup \Sigma_u$ avec Σ_c l'ensemble des événements contrôlables et Σ_u l'ensemble des événements incontrôlables.

Une définition plus détaillée est proposée dans [2].

C. Construction du masqueur

Pour obtenir le masqueur de fautes, il est nécessaire d'utiliser plusieurs modèles intermédiaires. Les méthodes de construction et calcul de ces modèles sont expliquées dans [9].

- Construire une version virtualisée du contrôleur. Ce modèle peut être obtenu automatiquement à partir du modèle du contrôleur supposé connu, et sert uniquement à la construction du masqueur. Cela revient à remplacer les événements de Σ par leur correspondant dans Σ_v (par exemple, c_g par c_{gv}).
- Déterminer les fautes possibles et établir le modèle d'accommodation aux fautes du procédé défaillant. Cela correspond au modèle du comportement nominal du procédé auquel on rajoute le modèle du comportement après occurrence de la faute.
- Établir la spécification de reconfiguration. Cette spécification assure que le masqueur ne va pas venir modifier les échanges entre le procédé et le contrôleur avant occurrence de la faute.

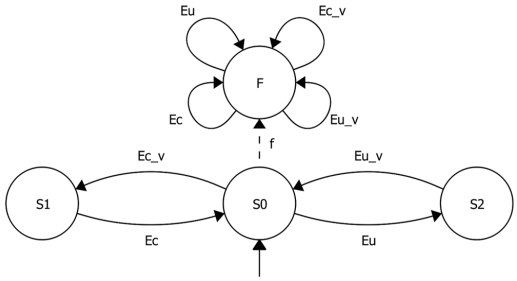

Fig. 3. Exemple de spécification de reconfiguration

La figure 3 expose un exemple de spécification de reconfiguration, avec $E_c \in \Sigma_c, E_u \in \Sigma_u, E_{cv} \in \Sigma_{cv}$ et $E_{uv} \in \Sigma_{uv}$. Avant occurrence de la faute f, l'émission d'un événement contrôlable réel (E_c) par le masqueur doit être précédée par l'émission de son correspondant virtuel (E_{cv}) par le contrôleur. De même, l'émission d'un événement virtuel incontrôlable (E_{uv}) doit être précédée par l'émission de son correspondant réel (E_v) par les composants du procédé. Après l'occurrence de f, correspondant à l'état F, le masqueur peut adapter librement les échanges.

- Établir la spécification d'accommodation aux fautes. Elle correspond au modèle du comportement attendu du procédé couplé au contrôleur, auquel on rajoute le comportement attendu après occurrence de la faute. Ce modèle est obtenu par expertise.

Pour obtenir le modèle du masqueur, on réalise la composition des modèles du contrôleur virtuel et du modèle d'accommodation aux fautes, auxquels on vient ajouter les spécifications de reconfiguration et d'accommodation. Les modèles de masqueur de fautes obtenus dans les deux cas de traitement de fautes ne sont pas présentés dans ce papier.

IV. Approche par reconfiguration du contrôle

A. Principe de la méthode de reconfiguration

La méthode présentée dans [3] propose une technique de reconfiguration du contrôleur, dont le principe est représenté à la figure 4. À l'aide d'un diagnostiqueur, les fautes sont détectées, isolées et rapportées au reconfigurateur. Le reconfigurateur va ensuite adapter la loi de contrôle.

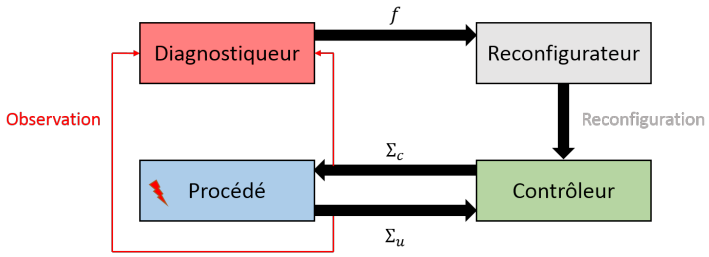

Fig. 4. Architecture d'un CTF avec *reconfigurateur du contrôle*

Cette méthode utilise le formalisme des AES et des modèles non-déterministes du procédé.

B. Automates Entrée/Sortie

Un *automate Entrée/Sortie*, noté G_{IO}, est défini par le 5-uplet

$$G_{IO} = (X, \Sigma_I, \Sigma_0, \delta, X_0) \tag{2}$$

avec X l'ensemble des états, Σ_I l'ensemble des événements d'entrée, Σ_O l'ensemble des événements de sortie, $\delta : X \times \Sigma_I \times \Sigma_O \rightarrow X$ la fonction de transition $\delta(x, e_i, e_o) = y$ signifiant qu'il existe une transition depuis l'état x qui sous occurrence de $e_i \in \Sigma_I$ va évoluer vers l'état y en générant l'événement $e_o \in \Sigma_O$ et X_0 l'ensemble possible d'états initiaux $X_0 \subset X$

La figure 5 expose une représentation graphique d'un AES.

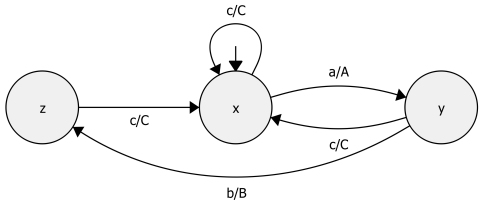

Fig. 5. Exemple d'un AES

Une définition plus détaillée est proposée dans [4].

C. Construction des modèles

La méthode de construction et de calcul de ces modèles est expliqué dans [3].

- Établir les modèles des composants du procédé. Cela correspond au comportement possible de l'ensemble des composants du procédé.
- Établir une spécification correspondant au comportement souhaité du système. Elle peut être représentée par une séquence d'états, un état final ou une séquence d'événements de sortie. Dans notre illustration, nous avons choisi d'utiliser une séquence d'événements de sortie pour notre spécification, qui représente les différentes combinaisons d'événements de sortie possibles tel que le chargement d'un petit colis c_p (respectivement d'un grand colis c_g) doit forcément être suivi du déchargement d'un colis à droite d_d (respectivement du déchargement d'un colis à gauche d_g), puis d'un retour en position centrale p_m de la table. Si cette séquence n'est pas unique, le nombre de séquences différentes correspond aux redondances des composants du procédé.
- Construire le treillis correspondant au modèle du procédé, puis déterminer les transitions et états vérifiant la spécification. Un treillis est un graphe représentant l'ensemble des états et des transitions actives d'un modèle, à partir de son ensemble d'états initiaux, puis à chaque pas de son évolution.

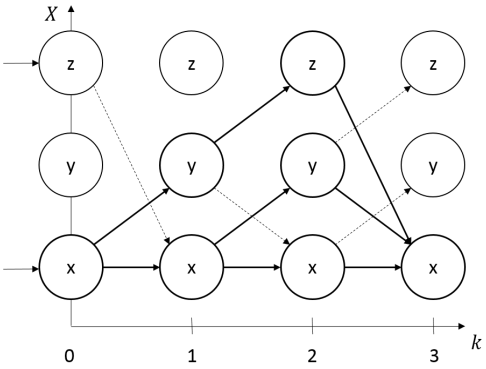

Fig. 6. Exemple de treillis

La figure 6 donne l'exemple d'un treillis obtenu à partir du modèle de la figure 5. Par soucis de lisibilité, les événements ne sont pas indiqués sur la figure.

- Extraire du treillis l'automate E/S à partir des états et transitions vérifiant la spécification (correspondant aux éléments en gras sur la figure 6).

Une fois le modèle du reconfigurateur obtenu, on peut extraire une loi de contrôle sans redondance représentant le fonctionnement du système en l'absence de faute décrit à la partie III.A.

La méthode propose ensuite une stratégie de reconfiguration du contrôleur en cas de détection d'une faute sur le système. Deux méthodes sont proposées :

- Replanification de trajectoire : on vient choisir dans le modèle du reconfigurateur une autre trajectoire permettant d'assurer le comportement du système.
- Adaptation d'entrée/sortie : l'événement d'entrée ou de sortie correspondant au composant défaillant est remplacé par un événement permettant d'assurer le comportement du système.

V. Traitement des différents types de fautes

Cette partie présente certains des modèles obtenus dans le cadre du traitement des fautes présentées dans la partie II.B. Par manque de place, seulement une partie des modèles est présentée. Ces modèles ont été choisis de manière à mettre en avant les limites et les points particuliers des deux méthodes.

A. Traitement du cas de fautes de type actionneur

La première faute prise en compte est la faute correspondant à la défaillance de l'actionneur commandant les rouleaux de la table dans le sens anti-horaire.

La figure 7 présente une partie de la spécification d'accommodation aux fautes des composants $\{table + rouleaux\}$. Par souci de taille des modèles, seulement une partie de la figure est présentée ici : la partie correspond à la prise en charge d'un petit colis. Le traitement d'un grand

colis fonctionne de manière similaire comme présenté en partie II.A.

Ce modèle construit par connaissance d'experts est com-

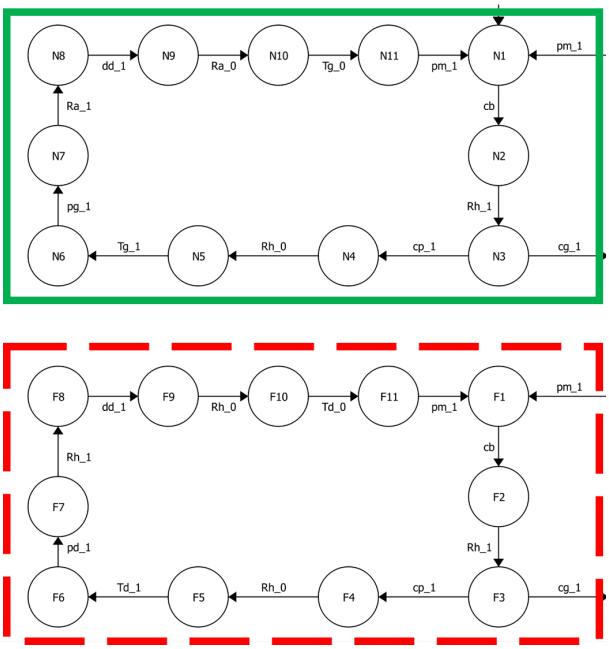

Fig. 7. Modèle de la spécification d'accommodation aux fautes dans le cas de la faute de type actionneur

posé de 2 parties : la partie délimitée par le cadre plein vert correspond au comportement nominal, alors que la partie délimitée par le cadre rouge en pointillés correspond au comportement souhaité après occurrence de la faute. Cela correspond respectivement aux stratégies de contrôle et de reconfiguration présentées en partie II.A et II.C. La figure 7 montre qu'il n'y a pas de transition allant du modèle du comportement nominal vers le modèle du comportement défaillant. Cela signifie qu'en l'absence de diagnostiqueur, il n'est pas possible de détecter l'occurrence de la faute par une séquence d'événements correspondant à un comportement défaillant. Le système reste donc bloqué dans un état d'attente d'événement (l'événement R_a_1).

La figure 8 présente le modèle du reconfigurateur obtenu pour le traitement de la faute de type actionneur. Ce modèle est obtenu à partir du treillis préalablement construit. Pour les mêmes raisons que la figure 7, seulement une partie de la figure 8 est présentée.

Il est possible d'extraire du modèle du reconfigurateur de la figure 8 une loi de contrôle respectant le comportement nominal, ce qui correspond ici au modèle de la figure 8 privé des états $\{7, 8, 9\}$. Après occurrence de la faute, cette dernière est rapportée au reconfigurateur, qui en utilisant la replanification de trajectoire, va permettre d'obtenir une nouvelle loi de contrôle correspondant au modèle de la figure 8 privé des états $\{4, 5, 6\}$.

B. Traitement du cas de fautes de type capteur

La deuxième faute prise en compte est la faute correspondant à la défaillance du capteur de fin de convoyeur A.

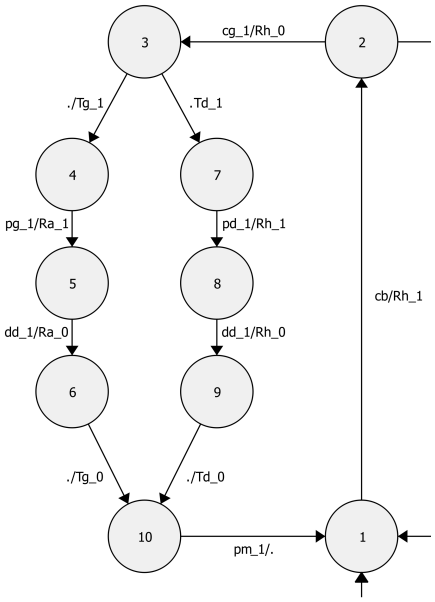

Fig. 8. Modèle du reconfigurateur pour la faute de type actionneur

La figure 9 présente une partie de la spécification d'accommodation aux fautes des composants $\{convoyeurA + convoyeurB\}$. De même que pour le premier cas de traitement de faute, le modèle de la figure 9 a été obtenu par connaissance d'experts, et les cadres ont la même signification. On peut remarquer sur la figure 9 qu'il existe une transition allant du modèle du comportement nominal au modèle du comportement défaillant (étiquetée par l'événement f). En effet, même sans diagnostiqueur, il est possible de détecter l'occurrence de la faute par une séquence d'événements fautive, qui ici correspond à l'occurrence d'un événement c_b_1 depuis l'état 0.

La figure 10 présente le modèle du reconfigurateur obtenu pour le traitement de la faute de type capteur. Ce modèle est obtenu à partir du treillis préalablement construit. Le modèle présent à la figure 10 étant sans redondance,

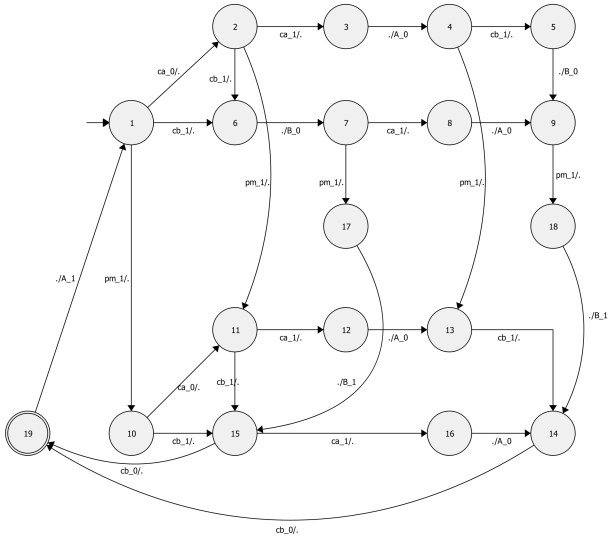

Fig. 10. Modèle du reconfigurateur pour la faute de type capteur

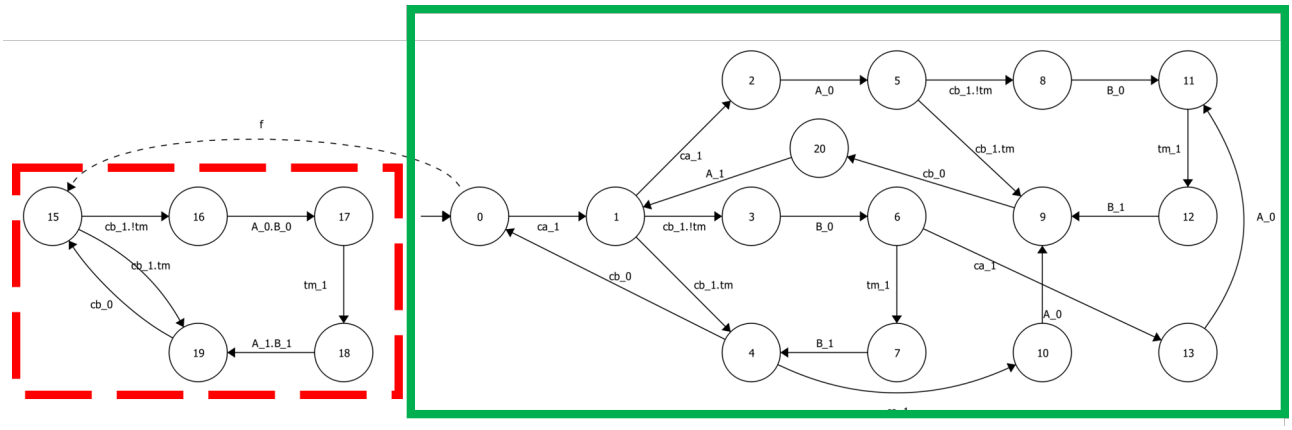

Fig. 9. Modèle de la spécification d'accommodation aux fautes dans le cas de la faute de type capteur

il correspond à la loi de contrôle respectant la spécification de contrôle. Après occurrence de la faute, le diagnostiqueur la rapporte au reconfigurateur. En utilisant l'adaptation d'entrée/sortie, il est alors possible de déterminer une nouvelle loi de contrôle. Dans le modèle de la figure 10, il faut alors modifier les transitions dans lesquelles apparaissent l'événement c_a, en remplaçant c_a_0 par 0 et c_a_1 par 1. Par exemple, la transition $\delta(1, c_a\_0, .) = 2$ devient $\delta(1, 0, .) = 2$.

VI. Discussion et conclusion

Concernant la capacité de traitement des différents types de fautes de l'approche par masquage de fautes, il n'a été possible d'obtenir un masqueur de fautes réalisant la commande tolérante aux fautes que dans un seul des cas d'application. En effet, pour le traitement de la faute de type actionneur, il n'est pas possible pour le masqueur de détecter l'occurrence de la faute, et donc d'adapter les échanges entre les composants du procédé et le contrôleur. Pour le deuxième type de faute en revanche, une séquence d'événements fautive permet de détecter la faute, et donc de réaliser la commande tolérante aux fautes à l'aide du masqueur. On peut aussi relever que la construction du masqueur nécessite la modélisation des fautes, ainsi que la modélisation du comportement du système après occurrence de la faute, ce qui expose la méthode à un problème d'explosion combinatoire lors d'un passage à l'échelle.

Pour ce qui est de la méthode de reconfiguration, il a été possible de réaliser la commande tolérante aux fautes dans les deux cas de fautes. A partir des modèles des reconfigurateurs, la replanification de trajectoire a permis d'obtenir une commande tolérante aux fautes utilisant les redondances du système dans le premier cas. Pour le deuxième type de fautes, l'utilisation de l'adaptation d'entrée/sortie a permis d'adapter la loi de contrôle de manière à réaliser le comportement dégradé après occurrence de la faute. Toutefois, les treillis nécessaires à la construction des reconfigurateurs ont été réalisés à la main car il n'existe pas d'outil permettant leur construction. Ce point en particulier est limitant car il rend impossible l'application de la méthode pour des systèmes dont la taille des modèles est plus importante.

Ces deux méthodes présentent des avantages et des inconvénients. La méthode par masquage de fautes [9] permet de réaliser une commande tolérante aux fautes sans diagnostiqueur. En contrepartie, elle n'est pas applicable à tous les types de fautes, et nécessite de modéliser les fautes et leurs répercussions sur le système. De l'autre côté, la méthode de reconfiguration du contrôle [3] est applicable à un plus grand nombre de fautes, mais nécessite l'utilisation d'un diagnostiqueur pour détecter l'occurrence de la faute. De plus, il n'existe pas d'outil facilitant l'application de la méthode.

De futurs travaux pourraient s'intéresser à l'application des méthodes sur le système complet, et non se limiter à une sous-partie du système. On pourrait aussi envisager la modélisation du comportement en prenant en compte l'occurrence de plusieurs types de fautes. Enfin, une extension de ces méthodes pourrait être envisagée afin de pouvoir répondre aux fautes de type processus, par exemple, en introduisant des formalismes prenant en compte le temps.

Références

[1] Blanke M., Kinnaert M., Lunze J. et Staroswieci M. *Diagnosis and Fault-Tolerant Control*. Springer, 2006.

[2] Cassandras C.G.et Lafortune S. *Introduction to Discrete Event Systems*. Springer, 2008.

[3] Nke Y. et Lunze J. Control reconfiguration based on unfolding of input/output automata. 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, SAFEPROCESS2012, Mexico, Mexique, 29-34 août 2012.

[4] Nke Y. Fault-Tolerant Control of Nondeterministic Input/Output Automata. Thèse de doctorat de l'Université de Bochum, 2008.

[5] Paoli A., Sartini M. et Lafortune S. A fault-tolerant architecture for supervisory control of discrete event systems. Proceedings of the 17th IFAC World Congress. Séoul, Corée du Sud, 6-11 juillet 2008.

[6] Ru Y. et Hadjicostis C. Fault-tolerant supervisory control of discrete event systems modeled by bounded petrinets. American Control Conference, ACC'07, New York, Etats-Unis, 9-13 juillet 2007.

[7] Wen Q. et Kumar R. A framework for fault-tolerant control of discrete event systems. *IEEE Transactions on Automatic Control*, vol. 53, iss. 8, pp. 1839-1849, septembre 2008.

[8] Wittmann T., Richter J. et Moor T. Fault-tolerant control of discrete event systems based on fault-accommodating models. 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, SAFEPROCESS2012, Mexico, Mexique, 29-34 août 2012.

[9] Wittmann T., Richter J. et Moor T. Fault-hiding control reconfiguration for a class of discrete event systems. 4th IFAC Workshop on Dependable Control of Discrete Systems, DCDS2013, York, Royaume-Uni, 4-6 septembre 2013.

[10] Moor T. *http ://www.rt.eei.uni-erlangen.de/FGdes/destool/*, 2008-2014.

[11] Real Games. *http ://www.realgames.pt/its-plc/*, 2004.