# Contrôle et extension des systèmes à événements discrets totalement et partiellement observables

**A. Khoumsi , G.v. Bochmann , R. Dssouli**

Université de Montréal
Faculté des arts et des sciences
Département d'informatique et
de recherche opérationnelle
C.P. 6128, Succursale A
Montréal, (Quebec)
H3C 3J7

RÉSUMÉ. Une méthode est proposée dans [7] pour générer d'une manière systématique un sous-module $M_2$ à adjoindre à un système à événements discrets $M_1$ afin de contrôler et d'étendre celui-ci. Le contrôle consiste à interdire certaines séquences d'événements de $M_1$, alors que l'extension consiste à ajouter de nouveaux événements à $M_1$. Dans [7], on fait l'hypothèse implicite qu'un événement de $M_1$ peut être interdit par $M_2$ si et seulement si il est observable par $M_2$. De plus, le comportement de $M_1$ en interaction avec $M_2$ peut contenir des états blocants. Dans cet article, nous avons supprimé l'hypothèse implicite en nous basant sur la théorie sur le contrôle développée par Ramadge et Wonham ([11]) et étendue par la suite par d'autres chercheurs. Dans la méthode que nous proposons, le comportement de $M_1$ en interaction avec $M_2$ ne contient pas d'état blocant. Nous proposons des algorithmes pour le calcul du module contrôleur $M_2$ dans le cas où le système à contrôler $M_1$ est totalement ou partiellement observable.

MOTS-CLÉS: systèmes à événements discrets, contrôleur, événement commandable, plus grand comportement commandable, observabilité, plus grand comportement T-observable, P-commandabilité, plus grand comportement commandable et T-observable.

## 1. Introduction

Un système à événements discrets (SED) est un système dynamique dans lequel les événements ont lieu instantanément, causant un changement discret de l'état du système ([11]). Dans cet article, nous considérons le cas où les séquences d'événements forment un langage régulier. Dans ce cas, la dynamique(ou le comportement) du SED peut donc être spécifiée par un automate à états finis où les transitions sont causées par les événements ([14]). Les SED se retrouvent dans plusieurs domaines. Un premier exemple de SED est un réseau d'ordinateurs ou de télécommunications. Dans ce cas, un événement est par exemple l'émission ou la réception d'un paquet de données. Un autre exemple de SED est un protocole de communications. Un événement est par exemple l'exécution d'une primitive d'établissement ou de relâchement de connexion. Un dernier exemple est un robot mobile accomplissant une tâche. Un événement peut être la détection d'un obstacle par un capteur ultrasonore.

Selon les applications, certaines séquences d'événements du SED ne doivent pas avoir lieu. Autrement dit, le SED doit être contrôlé de manière à avoir un certain comportement désiré ([4,7,11]). La synthèse de contrôleur consiste alors à dériver systématiquement le module contrôleur $M_2$ qu'il faut adjoindre au SED à contrôler $M_1$ afin d'obtenir un comportement le plus proche possible d'un certain comportement désiré. Soit donc un SED $M_1$ dont le fonctionnement est spécifié par un automate à états finis (AEF) $S_1$. Le but est de lui adjoindre un contrôleur $M_2$, de manière à ce que le système contrôlé $M_1 \| M_2$ (c.-à-d. $M_1$ et $M_2$ fonctionnant en parallèle et en interaction) fournisse un service spécifié par un AEF $S_0$ donné.

Ceci est schématisé sur la figure 1, où $K_1$, $K_2$, $K_3$ et $K_4$ sont des ensembles disjoints d'interactions et permettent de définir les vocabulaires (ou alphabets) des différents modules. En effet, si $V_0$, $V_1$ et $V_2$ sont respectivement les vocabulaires des modules $M_0$, $M_1$ et $M_2$, nous pouvons alors déterminer les $V_i$ à partir des $K_i$ par, $V_0=K_1 \cup K_2 \cup K_4$ , $V_1=K_1 \cup K_2 \cup K_3$ et $V_2 = K_2 \cup K_3 \cup K_4$. Nous pouvons aussi déterminer les $K_i$ à partir des $V_i$ par, $K_1=V_1-V_2$, $K_2=V_0 \cap V_1 \cap V_2$, $K_3= (V_1 \cap V_2)-V_0$ et $K_4=V_2-V_1$. $M_0$ est le module spécifié par le service $S_0$, qui est défini par l'ordonnancement des événements de $V_0$, càd de $K_1 \cup K_2 \cup K_4$ . $M_0$ est en fait la projection de $M_1 \| M_2$ (de vocabulaire $V=K_1 \cup K_2 \cup K_3 \cup K_4$) sur le vocabulaire $V_0$ . Remarquons aussi que $V_1 \cup V_2=V_1 \cup V_0=V_2 \cup V_0=V=K_1 \cup K_2 \cup K_3 \cup K_4$.

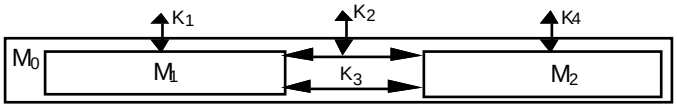



**Figure 1.** SED contrôlé

Signalons que $M_2$ n'est pas juste un contrôleur, car en plus de restreindre le comportement de $M_1$, il ajoute de nouveaux événements ($K_4$ sur la figure 1). Nous le qualifions ainsi de contrôleur étendu. Si nous considérons un réseau de télécommunications, le contrôle est nécessaire pour la gestion du réseau. Considérons deux exemples très simples.

**Exemple 1:** Soient deux noeuds d'un réseau reliés par deux lignes, et où le choix de l'une des deux lignes est fait à l'aide d'un commutateur. La première ligne est utilisée en fonctionnement normal. Si celle-ci tombe en panne, il faut commuter automatiquement sur la seconde ligne. Si la première ligne est réparée, il faut commuter automatiquement sur celle-ci. Pour réaliser cette commutation, il faut disposer d'un contrôleur qui détecte les événements "ligne 1 en panne" et "ligne 1 réparée", et qui déclenche les événements "commute sur la ligne 1" et "commute sur la ligne 2".

**Exemple 2:** Il s'agit d'un contrôleur d'un robot mobile devant rejoindre une zone donnée et éviter les obstacles éventuels qui peuvent se trouver sur son chemin. Un tel contrôleur doit être informé des événements tels que "capteur ultrasonore a détecté obstacle gênant" et "robot arrivé à destination", et il doit pouvoir déclencher les événements tels que "va au but", "arrête" et "commence contournement d'obstacle" ...

Dans les deux exemples ci-dessus, le contrôleur pourrait être conçu intuitivement. Mais dans des cas plus complexes, il est indispensable de disposer d'une méthode rigoureuse et systématique permettant la synthèse du contrôleur $M_2$. Dans [7], une méthode est proposée pour calculer à l'aide d'une formule le module contrôleur $M_2$. Mais les deux hypothèses suivantes sont faites : a) $M_2$ n'a aucun effet sur les événements de $M_1$ qu'il ne peut pas observer; b) $M_2$ contrôle tous les événements de $M_1$ qu'il peut observer. Ces hypothèses, assez restrictives, impliquent que sur la figure 1, $K_1$ correspond aux événements non observables par $M_2$, et $K_2$ et $K_3$ sont les événements observables par $M_2$. Dans [11] ces hypothèses ne sont pas faites, mais $K_3$ et $K_4$ sont supposés vides, càd $V_2 \subseteq V_1 = V_0$. Nous reprenons l'étude de calcul du module $M_2$ en supprimant les hypothèses faites dans [7] et celles faites dans [11].
Nous allons montrer qu'on peut aisément reprendre l'étude du contrôle ([11]) dans le cas plus général où $K_3$ et/ou $K_4$ ne sont pas vides. Dans la section 2, nous donnons différentes définitions. Dans la section suivante, le problème du contrôle et la réduction de celui-ci sont présentés. Dans la section 4, nous étudierons le contrôle des systèmes totalement observables. Nous présentons d'une manière intuitive puis formelle le principe de calcul du comportement commandable le plus grand (CCPG). L'algorithme que nous présentons n'est pas équivalent à celui décrit dans [14] et qui considère $K_3=K_4=\emptyset$, mais il utilise également la méthode du point fixe et il est de même complexité dans le pire cas. Dans la section 5, nous étudierons le contrôle des systèmes partiellement observables. Nous proposons, dans un premier temps, un algorithme utilisant la formule Merlin-Bochmann ([7]) pour calculer le comportement T-observable le plus grand (CTPG). Nous proposons ensuite, un algorithme permettant de calculer le comportement commandable et T-observable le plus grand (CCTPG). Cet algorithme est inspiré de la procédure proposée dans [3] et qui considère aussi $K_3=K_4=\emptyset$. Signalons que les différences de nos algorithmes avec ceux de [3,14] ne sont pas dues seulement à l'hypothèse $K_3=K_4=\emptyset$. Dans la section 4 (resp. 5), la spécification du contrôleur $M_2$ est déduite du CCPG (resp. CCTPG). Et enfin dans la section 6, nous concluons et proposons quelques extensions. Avant d'aller plus loin, voici tout d'abord une table des principales notations utilisées.

| | |
|---|---|
| SED | signifie : Système à événements discrets. |
| AEF (resp. AEFM) | signifie : Automate à états finis (resp. Automate à états finis marqué). |
| CD | signifie : Comportement désiré. |
| CCPG (resp. CTPG) | signifie : Comportement commandable (resp. T-observable) le plus grand. |
| CCTPG | signifie : Comportement commandable et T-observable le plus grand. |
| $(Q, V, \delta, q_0)$ définit un AEF sur le vocabulaire (ou l'alphabet) V. | |
| $n_i$ est le nombre d'états d'un AEF $S_i$. | |
| $(Q,V,\delta,q_0,Q_m)$ définit un AEF, dit marqué, possédant un ensemble $Q_m$ d'états marqués, avec $Q_m \subseteq Q$. | |
| $\delta(q,\sigma)$ est l'état atteint à partir de l'état q, après l'occurrence de l'événement $\sigma$. | |
| $\delta(q,\sigma)!$ signifie que $\delta(q,\sigma)$ est défini. $\delta(q,\sigma)\neg!$ est la négation de $\delta(q,\sigma)!$. | |
| pr(T) est l'ensemble des séquences préfixes d'une séquence T d'événements. | |
| $\delta(q,T)$ est l'état atteint à partir de l'état q après exécution de la séquence T. Formellement, $\delta(q,T)! \Leftrightarrow \forall\ s \in pr(T): \delta(q,s)!$. | |
| $V_i$ est le vocabulaire (ou alphabet) d'un module $M_i$ spécifié par un AEF $S_i$. | |
| $V_i^*$ est l'ensemble des séquences finies des événements de l'alphabet $V_i$. | |
| $\underline{S}$ est un AEF marqué défini à partir d'un AEF S. Il accepte toutes et seulement les séquences non acceptées par S. | |
| $P_i(S)$ (resp. $P_i(s)$) est la projection sur l'alphabet $V_i$ du AEF S (resp. de la séquence s d'événements). | |
| $S_i \times S_j$ est le produit synchronisé de deux AEFs $S_i$ et $S_j$, (synchronisation sur les événements de $V_i \cap V_j$). | |
| $S_i \otimes S_j$ est obtenu à partir de $S_i \times S_j$ en supprimant tous les états absorbants (déf.1) de $S_i \times S_j$. | |
| $M_1 \| M_2$ représente deux modules $M_1$ et $M_2$ fonctionnant en parallèle et éventuellement en interaction. | |
| $V_{cd}$ (resp. $V_{nc}$) est l'ensemble des événements commandables (resp. non commandables) par $M_2$, avec $V_{cd} \subseteq V_2$. | |
| $V_{ob}$ (resp. $V_{no}$) est l'ensemble des événements observables (resp. non observables) par $M_2$, avec $V_{ob} \subseteq V_2$. | |
| $S_c$ (resp. $S_{to}$, $S_{ct}$) est la spécification du CCPG (resp. CTPG, CCTPG). | |

**Table 1.** Notations

**2. Définitions**

**Définition 1.** (*AEF, événement tirable, état blocant, et état absorbant*).
Un automate à états finis (AEF) est défini par $(Q,V,\delta,q_0)$. Q est l'ensemble des états, V est l'alphabet, $\delta$ est une fonction partielle définissant les transitions, c.-à-d. $\delta: Q\times V\rightarrow Q$, et $q_0$ est l'état initial.
Pour un système spécifié par un AEF $(Q,V,\delta,q_0)$, un événement $\sigma$ est tirable, si son ocurrence est possible à partir de l'état courant du système. Formellement, si q est l'état courant : $\sigma$ est tirable $\Leftrightarrow \delta(q,\sigma)!$ .
Un état q d'un AEF défini par $(Q,V,\delta,q_0)$ est dit *blocant (deadlock)* si aucune transition n'est tirable de cet état. Formellement : q est bocant $\Leftrightarrow \forall\, \sigma\in V, \delta(q,\sigma)\neg!$ .
Un état q d'un AEF défini par $(Q,V,\delta,q_0)$ est dit *absorbant,* s'il est blocant ou si toute transition tirable à partir de q, n'entraîne pas un changement d'état. Formellement : q est absorbant $\Leftrightarrow \forall\, \sigma\in V, (\delta(q,\sigma)\neg!) \vee (\delta(q,\sigma)=q)$. ■

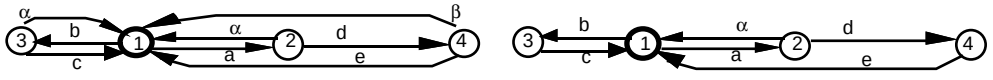
**Définition 2.** (*AEF marqué, AEF correspondant, premier et dernier états marqués*).
Un AEF marqué (AEFM) $S_m$ est défini par $(Q,V,\delta,q_0,Q_m)$, où $(Q,V,\delta,q_0)$ définit un AEF dit correspondant à $S_m$, et $Q_m$ est un sous-ensemble de Q d'états marqués. Les états de $Q_m$ sont identifiés par des nombres. L'état identifié par le plus petit (resp. grand) nombre, est appelé premier (resp. dernier) état de $Q_m$. ■

**Définition 3.** (*Séquence acceptée, équivalence et relation d'ordre partiel entre AEFs et AEFMs*)
- Soit $S_1=(Q_1,V_1,\delta_1,q_{10})$ et $S_2=(Q_2,V_2,\delta_2,q_{20})$ deux AEFs. Une séquence s *acceptée* par un AEF $S_i$ est notée $s\in S_i$ .
- $S_1$ et $S_2$ sont *équivalents*, et sont notés $S_1\cong S_2$, si et seulement s'ils acceptent le même langage régulier.
Formellement : $S_1\cong S_2 \Leftrightarrow (V_1=V_2)\wedge(\forall\, s\in V_1^{*} : \delta_1(q_{10}, s)! \Leftrightarrow \delta_2(q_{20}, s)!)$
- $S_1$ est *plus petit que* $S_2$, ce qui est noté $S_1\leq S_2$, si et seulement si le langage accepté par $S_1$ est inclus dans celui accepté par $S_2$. Dans ce cas, nous pouvons aussi dire que $S_2$ est *plus grand que* $S_1$. Formellement :
$S_1\leq S_2 \Leftrightarrow (V_1\subseteq V_2)\wedge(\forall\, s\in V_1^{*} : \delta_1(q_{10}, s)! \Rightarrow \delta_2(q_{20}, s)!)$. Pour la relation $\leq$, $S_1$ et/ou $S_2$ peuvent être des AEFMs. Dans ce cas, on considère simplement les AEFs correspondants (déf.2) pour effectuer la relation. ■

**Définition 4.** (*Opérateur $Comp_V$*)
Soit V un alphabet donné, et soit $S_i$ un AEF quelconque défini par $(Q_i,V_i,\delta_i,q_{i0})$, avec $V_i\subseteq V$. $Comp_V(S_i)$ est obtenu en ajoutant à chaque état de $S_i$ , tous les événements de $V\text{-}V_i$ . Ces derniers n'entraînent pas un changement d'état de $S_i$. $Comp_V(S_i)$ est en fait l'AEF le plus grand (déf.3) de vocabulaire V et dont la projection sur $V_i$ , est équivalente (déf.3) à $S_i$. Formellement : $Comp_V(S_i)=(Q_i,V,\delta_{Vi},q_{i0})$ , avec : $\forall\, q\in Q_i : (\sigma\in V_i \Rightarrow \delta_{Vi}(q,\sigma)=\delta_i(q,\sigma))\wedge(\sigma\in V\text{-}V_i \Rightarrow \delta_{Vi}(q,\sigma)=q)$. ■

**Définition 5.** (*Événement commandable, ensembles $V_{cd}$ et $V_{nc}$* )
Soit un système à contrôler $M_1$ spécifié par $S_1=(Q_1,V_1,\delta_1,q_{10})$ et relié à un contrôleur $M_2$, (sect.1, fig.1). Un événement $\sigma$ de $V_1$ est *non commandable* s'il ne peut pas être interdit par $M_2$ à partir de tout état de $S_1$ d'où il est tirable(déf.1). Si l'événement $\sigma$ peut être interdit, il est dit *commandable* ([10]). Autrement dit, $M_2$ ne peut influer que sur les occurrences des événements commandables. Nous définissons alors $V_{cd}$ et $V_{nc}$ qui sont respectivement les ensembles des événements commandables et non commandables. Les événements ajoutés par $M_2$, ($K_4=V_2\text{-}V_1$, fig.1), sont commandables. Donc, $K_4\subseteq V_{cd}$ et $V_{nc}\subseteq V_1$. Les événements avec lesquels $M_2$ n'a aucune interaction, ($K_1=V_1\text{-}V_2$, fig.1) ne sont pas commandables. Donc, $K_1\subseteq V_{nc}$ et $V_{cd}\subseteq V_2$. ■

**Exemple 2:** sur la figure 2, a,b,c,d et e appartiennent à $V_{cd}$, $\alpha$ et $\beta$ appartiennent à $V_{nc}$. Il n'est alors pas possible de restreindre le comportement de $M_1$ spécifié par $S_1$, pour que le système contrôlé (c.-à-d. $M_1\|M_2$) soit spécifié par $S_0$. En effet, cela nécessiterait d'interdire respectivement les événements non commandables $\alpha$ et $\beta$ à partir des états 3 et 4.




**2.a.** $S_1$ **2.b.** $S_0$

**Figure 2**. Comandabilité des événements

**Définition 6.** (*Événement observable*)
Un événement de $M_1$ est *observable* par $M_2$ si son occurrence peut être détectée par $M_2$. Nous définissons alors $V_{ob}$ et $V_{no}$ qui sont respectivement les ensembles des événements observables et non observables. Les événements ajoutés par $M_2$, c.-à-d. $K_4=V_2\text{-}V_1$ (sect.1, fig.1) sont observables. Donc, $K_4\subseteq V_{ob}$ et $V_{no}\subseteq V_1$. Les événements avec lesquels $M_2$ n'a aucune interaction, ($K_1=V_1\text{-}V_2$, fig.1) ne sont pas observables. Donc, $K_1\subseteq V_{no}$ et $V_{ob}\subseteq V_2$. ■

Nous déduisons des definitions 5 et 6 et de la figure 1 que, $K_1=V_{no}\cap V_{nc}$, $V_2=K_2\cup K_3\cup K_4=V_{ob}\cup V_{cd}$ et $K_4\subseteq V_{ob}\cap V_{cd}$ . Dans [7], un événement est commandable si et seulement si il est observable. Dans ce cas particulier, nous avons alors $V_{ob}=V_{cd}$ , et $K_1=V_{no}=V_{nc}$ et $V_2=K_2\cup K_3\cup K_4=V_{ob}=V_{cd}$. Cette hypothèse est supprimée dans notre article. Une manière de représenter schématiquement la commandabilité et l'observabilité des événements de $M_1$ par $M_2$ est la suivante: a) un événement de $V_{cd}\cap V_{no}$ est représenté par : ◄—●—► ; b) un événement de $V_{nc}\cap V_{ob}$ est représenté par : ◄—○—► ; c) un événement de $V_{cd}\cap V_{ob}$ est représenté par : ◄——►.

**Remarque :** seuls les événements communs à $M_1$ et $M_2$ (c.-à-d. les événements de $K_2 \cup K_3$) sont concernés par cette représentation. Pour $K_1$ et $K_4$, nous rappelons que $K_1 = V_{no} \cap V_{nc}$ et $K_4 \subseteq V_{ob} \cap V_{cd}$ .

**Exemple 3:** (fig. 3) $b \in V_{no} \cap V_{cd}$, $c \in V_{ob} \cap V_{nc}$, $d \in V_{ob} \cap V_{cd}$, et bien sûr $a \in V_{no} \cap V_{nc}$ et $e \in V_{cd} \cap V_{ob}$.

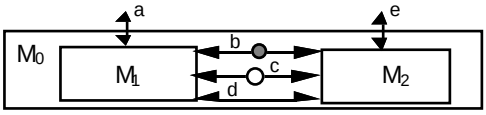



**Figure 3.** Événements commandables et observables

**Définition 7.** (*Contrôleur $M_2$*)
Intuitivement, $M_2$ est un système discret à adjoindre à $M_1$ et qu'on peut qualifier d'observateur actif. $M_2$ est un observateur, car il détecte les occurrences de tous les événements appartenant à $V_{ob}$ (déf.6). $M_2$ est actif, car il peut à tout moment interdire l'occurrence de tout événement appartenant à $V_{cd}$ (déf.5). En plus, $M_2$ ajoute de nouveaux événements ($K_4$ sur la figure 1) à $M_1$. Autrement dit, dans le but d'atteindre un certain comportement désiré, $M_2$ observe l'évolution de $M_1$, et met à jour les événements de $V_{cd}$ qu'il autorise. Formellement, $M_2$ est spécifié par $(S_2, \Psi)$ où : a) $S_2=(Q_2, V_{ob}, \delta_2, q_{20})$ est un AEF de vocabulaire $V_{ob}$ ; b) $\Psi$ est une fonction qui associe à tout état de $S_2$, l'ensemble des événements qui sont autorisés par $M_2$. Nous avons bien sûr : $\forall\ q \in Q_2,\ V_{nc} \in \Psi(q)$. ■

Dorénavant, le terme comportement désignera des comportements sans aucune projection. C'est-à-dire, un comportement est spécifié par un AEF de vocabulaire $V_1 \cup V_2$.

**Définition 8.** (*Comportement commandable*)
Soit $M_1$ un système à contrôler spécifié par $S_1=(Q_1, V_1, \delta_1, q_{10})$, et soit alors $Comp_V(S_1)=(Q_1, V, \delta_{V1}, q_{10})$ (déf.4), avec $V=V_1 \cup V_2=V_0 \cup V_2$. Soit $S_c=(Q_c, V, \delta_c, q_{c0})$ un AEF tel que $S_c \leq Comp_V(S_1)$ (déf.3). $S_c$ et le comportement qu'il spécifie sont commandables, si et seulement si les trois conditions suivantes sont vérifiées.

C1 : après l'occurrence d'une séquence t d'événements acceptée par $S_c$ et par $Comp_V(S_1)$, un événement non commandable (déf.5) tirable dans $Comp_V(S_1)$, n'est pas interdit dans $S_c$ ,
C2 : tout état q de $S_c$ n'est pas un état absorbant (et donc pas blocant) (déf.1),
C3 : tout état q de $S_c$ est atteignable à partir de l'état initial de $S_c$ .

Plus formellement les trois conditions sont les suivantes:

C1 : $\forall\ t \in V^*, \forall\ \alpha \in V_{nc} : ((\delta_{V1}(q_{10}, t)!) \wedge (\delta_c(q_{c0}, t)!) \wedge (\delta_{V1}(q_{10}, t\alpha)!)) \Rightarrow \delta_c(q_{c0}, t\alpha)!$
C2 : $\forall\ q \in Q:\ \exists\ \sigma \in V$, tel que $(\delta_c(q,\sigma)!) \wedge (\delta_c(q,\sigma) \neq q)$
C3 : $\forall\ q \in Q:\ \exists\ s \in V^*$, tel que $q = \delta_c(q_{c0}, s)$

Intuitivement, C1 est une condition obligatoire. En effet, si elle n'est pas respectée, alors $S_c$ n'est pas réalisable. En effet, $S_c$ ne peut dans ce cas être réalisé qu'en interdisant des événements non commandables. La condition C2 est arbitraire, elle impose à un comportement commandable d'être "dynamique" , c.-à-d. de pouvoir toujours changer d'état. Et enfin, la condition C3 impose à $S_c$ de ne contenir que les états significatifs, c.-à-d. éventuellement atteignables.
Signalons que notre définition est différente de la définition traditionnelle ([14]) où seules les conditions C1 et C2 sont nécessairement vérifiées. ■

Sur l'exemple 2 (fig.2), le comportement $S_0$ n'est pas commandable, car la condition C1 n'est pas vérifiée. En effet, $S_0$ nécessite d'interdire les événements non commandables $\alpha$ et $\beta$ à partir respectivement des états 3 et 4.

### 3. Présentation et réduction du problème du contrôle des SED

#### 3.1. Problème du contrôle

**Définition 9.** (*Comportement désiré*)
Lorsque le système $M_1$ est libre, son comportement est spécifié par un AEF $S_1$. Le but du contrôle est d'adjoindre à $M_1$ un module contrôleur $M_2$ (déf.7), afin que le système $M_1 \| M_2$ (Table 1) fournisse un certain service désiré spécifié par un AEF $S_0$ sur l'alphabet $V_0$. Pour que $M_1 \| M_2$ fournisse le service en question, il faudrait que la spécification S de son comportement défini sur l'alphabet $V=V_1 \cup V_0$, soit telle que $S \leq S_1 \times S_0$ (déf.3). $S_1 \times S_0$ spécifie donc le comportement désiré (ou simplement CD) qu'il faudra essayer d'obtenir, et sera désigné par $S_{cd}=S_1 \times S_0$. ■

Pour, la conception de $M_2$, considérons deux cas.
*Cas 1* : $V_{no}=\varnothing$. Donc, les occurrences de tous les événements de $M_1$ sont détectées par $M_2$. Si le comportement désiré (CD) n'est pas commandable (déf.8), le but est alors de calculer la spécification $S_c$ du comportement commandable le plus grand (déf.3), noté CCPG, qui soit plus petit que le CD (déf.9). Il faut ensuite calculer la spécification $(S_2, \Psi)$ du contrôleur $M_2$ (déf.7) permettant de réaliser le CCPG , c.-à-d. que $M_1 \| M_2$ sera spécifié par $S_c$.
*Cas 2* : $V_{no} \neq \varnothing$. Lors de l'évolution du système, le contrôleur met à jour l'ensemble des événements autorisés. Cet ensemble dépend de l'état courant de $M_1$. Si les occurrences de certains événements, dits non observables (déf.6), ne sont pas détectées par $M_2$, celui-ci ne peut pas connaître en permanence l'état du système, qualifié de partiellement

observable, et ne peut alors pas en général décider quelles sont les événements commandables qu'il doit interdire. Dans ce cas, le CD (déf.9) n'est pas réalisable, même s'il est commandable ([11,14]). Il faut donc, à partir d'un CD, et compte tenu de la non commandabilité et de la non observabilité de certains événements, calculer la spécification $S_{ct}$ du comportement réalisable le plus grand, noté CCTPG, qui soit plus petit que le CD et qui vérifie une certaine propriété. Il faut ensuite calculer la spécification $(S_2, \Psi)$ du contrôleur $M_2$ permettant de réaliser le CCTPG , c.-à-d. que $M_1 \| M_2$ sera spécifié par $S_{ct}$ ([2,3,6,11,13]). Les deux cas sont traités en détail, respectivement dans les sections 4 et 5.

### 3.2. Réduction du problème du contrôle

Dans le cas général, [7] considère que $K_3$ et $K_4$ ne sont pas vides. Mais le problème du contrôle peut être réduit à un problème où $K_3$ et $K_4$ sont vides, c.-à-d. où les vocabulaires de $S_1$ et $S_0$ sont égaux. Pour cela, il faut transformer $S_1$ en $S_{p1}=Comp_V(S_1)$ et $S_0$ en $S_{p0}=Comp_V(S_0)$, où $V=V_1 \cup V_0$. $S_{p1}$ et $S_{p0}$ ont le même vocabulaire $V=V_1 \cup V_0$, et les modules $M_{p1}$ et $M_{p0}$ respectivement spécifiés par $S_{p1}$ et $S_{p0}$ sont schématisés sur la figure 4 , avec $K=K_2 \cup K_3 \cup K_4$. On peut montrer que le problème du contrôle qui est initialement de déterminer $(S_2, \Psi)$ à partir de $S_1$ et $S_0$ , peut donc se réduire à déterminer le même $(S_2, \Psi)$ à partir de $S_{p1}$ et $S_{p0}$.

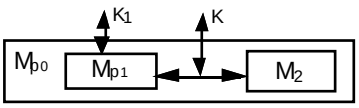



**Figure 4.** Réduction du problème

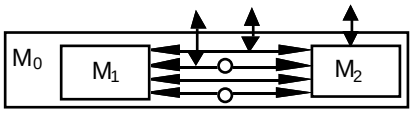



**Figure 5.** Système contrôlé totalement observable

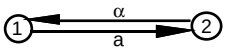
### 4. Contrôle des systèmes discrets totalement observables ($V_{no} = \emptyset$)

Dans cette section, nous supposons $V_{no} = \emptyset$. Soit donc $V_1 \cup V_2 = V_2 = V_{ob} = V_{cd} \cup V_{nc}$ (fig. 5).

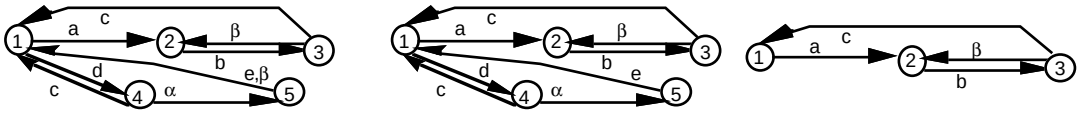
### 4.1 Calcul du comportement commandable le plus grand (CCPG)

Soit $M_1$ un système à contrôler de vocabulaire $V_1$ spécifié par $S_1=(Q_1,V_1,\delta_1,q_{10})$, et soit le service désiré $S_0=(Q_0,V_0,\delta_0,q_{00})$. Le comportement désiré est donc spécifié par $S_{cd}=S_1 \times S_0$ (déf.9). Dans le cas présent où $V_{no}=\emptyset$, tout comportement commandable (déf.8) est réalisable ([14]). Le but (sect.3.1) est alors de calculer la spécification $S_c$ du comportement commandable le plus grand (CCPG) telle que $S_c \leq S_{cd}$. Si $S_{cd}$ est commandable, alors $S_c \cong S_{cd}$. Sinon, $S_c \lhd S_{cd}$.

**Remarque :** du fait de la condition C2 (déf.8), le CCPG $S_c$ est non seulement plus petit que $S_1 \times S_0$, mais aussi plus petit que $S_1 \otimes S_0$ (Table 1). De plus, si $S_{cd}=S_1 \times S_0$ est commandable, alors $S_{cd}=S_1 \otimes S_0$ .

Nous présentons dans la suite comment obtenir la spécification $S_c$ du CCPG à partir $S_1$ et $S_0$, et connaissant $V_1$, $V_2$, $V_0$ et $V_{cd}$. Un algorithme de calcul du CCPG est décrit formellement dans [14]. Celui que nous proposons est différent, mais il utilise lui aussi la méthode du point fixe et il est de même complexité dans le pire cas. De plus, il est adapté au cas général où $K_3$ et/ou $K_4$ sont non vides. Nous expliquons tout d'abord la méthode d'obtention de $S_c$ d'une manière informelle en utilisant deux exemples (figures 2 et 6).
Sur l'exemple de la figure 2, comme il n'est pas possible d'interdire les événements $\alpha$ et $\beta$ à partir des états 3 et 4, ces derniers doivent être évités. L'événement b ne doit donc pas être tiré de l'état 1, de même que l'événement d ne doit pas être tiré de l'état 2. Ceci ne pose aucun problème car les événements b et d sont commandables et peuvent donc être interdits par le contrôleur. Le CCPG est donc spécifié par $S_c$ suivant :  .


Sur la figure 6, avec $V_1=V_0=\{a,b,c,d,e,\alpha,\beta\}$, $V_{cd}=\{a,b,c,d,e\}$ et $V_{nc}=\{\alpha,\beta\}$, $S_0 \times S_1$ ($=S_0$ car $V_0=V_1=V_2$) n'est pas commandable car il n'est pas possible d'interdire $\beta$ à partir de l'état 5. Ce dernier doit donc être évité, et pour cela l'événement $\alpha$ partant de l'état 4 ne doit pas être tiré. Comme $\alpha$ ne peut pas être interdit, l'état 4 doit être évité. Pour cela l'événement d, partant de l'état 1, ne doit pas être tiré.




**6.a.** $S_1$ **6.b.** $S_0$ **6.c.** $S_c$

**Figure 6.** Comportement non commandable

Le comportement commandable obtenu est spécifié par $S_c$ (fig.6.c). Pour définir formellement la méthode d'obtention de $S_c$, nous introduisons trois opérateurs $C_A$ et Ct, et $Cont_A$. Ce dernier opérateur est défini à partir des deux premiers.

**Définition 10.** (*Opérateur $C_A$*)
Intuitivement, soient A et B deux AEFs de vocabulaires $V_A$ et $V_B$. $C_A(B)$ est l'AEFM (déf.2) le plus grand tel que

$C_A(B) \leq A \times B$, dont tout état est accessible à partir de l'état initial, et tel que tout état $e = \langle e_A, e_B \rangle$ de $C_A(B)$ qui vérifie une des deux conditions ci-après, est marqué et ne possède pas de transition sortante (c.-à-d. un état marqué est blocant). Condition 1: e est un état absorbant de $A \times B$. Condition 2: il existe au moins un événement p non commandable tel que p est tirable dans A, à partir de l'état $e_A$, et p n'est pas tirable dans B, à partir de $e_B$.
L'opérateur $\mathbf{C_A}$ est défini plus formellement comme suit. Soient A et B les AEFs $(Q_A, V_A, \delta_A, q_{A0})$ et $(Q_B, V_B, \delta_B, q_{B0})$, et soient alors $Comp_V(A)$ et $Comp_V(B)$ (déf.4) les AEFs $(Q_A, V_A \cup V_B, \delta_{VA}, q_{A0})$ et $(Q_B, V_A \cup V_B, \delta_{VB}, q_{B0})$, avec $V = V_A \cup V_B$.
$C_A(B)$ est alors l'AEFM $(Q, V_A \cup V_B, \delta, q_0, Q_m)$ tel que : (a) $\forall\ T \in (V_A \cup V_B)^* : \delta(q_0, T)! \Leftrightarrow P1 \wedge P2$ ;
(b) $\forall\ q \in Q : (\forall\ s \in V_A \cup V_B,\ \delta(q,s) \neg! \vee \delta(q,s) = q) \Leftrightarrow (q \in Q_m)$.
Avec $P1 = \delta_{VA}(q_{A0}, T)! \wedge (\delta_{VB}(q_{B0}, T)!$ et $P2 = \forall t \in pr(T) - \{T\}, \forall\ \sigma \in V_{nc} : \delta_{VA}(q_{A0}, t\sigma)! \Rightarrow \delta(q_0, t\sigma)!$ .
P1 indique que $C_A(B) \leq A \times B$, P2 indique que l'interdiction par $C_A(B)$ d'un événement non commandable autorisé par A, ne se fait jamais sur un état non absorbant de $C_A(B)$, et (b) indique que les états marqués sont absorbants. ■

Si nous reprenons l'exemple de la figure 6, nous obtenons $C_{S1}(S_0)$ de la figure 7 ($Q_m$={état 5}). L'état 5 est marqué car l'événement β est possible sur $S_1$ et interdit sur $S_0$. Un état marqué est en fait un état à éviter. Une fois $C_{S1}(S_0)$ obtenu, une méthode itérative utilisant un second opérateur Ct permet de calculer $S_c$.

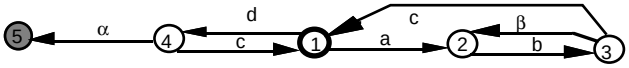



**Figure 7.** L'opérateur $C_{S1}(S_0)$

**Remarque:** $C_{S1}(S_0) = C_{S1}(Comp_V(S_0))$. Si $S = S_0$ est de vocabulaire V avec $V_1 \subseteq V$ et est commandable, alors $C_{S1}(S) = S$.

**Définition 11.** (*Opérateur Ct*)
Si A est un AEFM défini par $(Q_A, V_A, \delta_A, q_{A0}, Q_{mA})$, alors **Ct(A)** est l'AEFM défini par $(Q, V_A, \delta, q_0, Q_m)$ tel que, si $Q_{mA} = \varnothing$ alors Ct(A)=A. Sinon (c.-à-d. $Q_{mA} \neq \varnothing$), soit $q_{mA}$ le premier état (déf.2) de $Q_{mA}$, Ct(A) est alors tel que :
a) $Q = Q_A - \{q_{mA}\}$; b) $\forall\ q \in Q, \forall\ s \in V_A : \delta(q,s)! \Leftrightarrow (\delta_A(q,s)! \wedge (\delta_A(q,s) \neq q_{mA})) \Rightarrow (\delta(q,s) = \delta_A(q,s))$;
c) $\forall\ q \in Q : q \in Q_m \Leftrightarrow (C'1 \vee C'2 \vee C'3 \vee C'4)$. Avec $C'1 = (\exists\ \sigma \in V_{nc}$ tel que : $\delta_A(q, \sigma) = q_{mA})$,
$C'2 = (\forall\ s \in V_A : (\delta_A(q,s) = q_{mA}) \vee (\delta_A(q,s) \neg!) \vee (\delta_A(q,s) = q))$, $C'3 = (\forall\ s \in V_A, \forall r \neq q_{mA} : \delta_A(r,s) \neq q)$, et $C'4 = (q \in Q_{mA} - \{q_{mA}\})$.
Intuitivement, Ct(A) consiste à premièrement supprimer le premier état marqué de A, et donc les transitions qui y mènent directement, et celles qui sont tirables de cet état. Cette suppression peut engendrer des états interdisant des événements non commandables (C'1), des états absorbants (C'2), ou des états non atteignables à partir de l'état initial (C'3). Ces états sont alors marqués pour donner l'automate Ct(A). Nous déduisons, d'après la définition 8, que si Ct(A) ne possède pas d'états marqués, alors Ct(A) est commandable. La réciproque est aussi vraie. ■

**Définition 12.** (*Opérateur $Cont_{S1}$*)
L'opérateur $Cont_{S1}$ consiste à appliquer une fois l'opérateur $C_{S1}$ sur $S_0$, et itérativement l'opérateur Ct jusqu'à obtenir un automate à états finis sans états marqués. Soit donc $\mathbf{Cont_{S1}(S_0)}$ défini algorithmiquement par :

**$Cont_{S1}(S_0)$** *Début*: - $A \leftarrow C_{S1}(S_0) = (Q_A, V_A, \delta_A, q_{A0}, Q_{mA})$
- *Tant que* $Q_{mA} \neq \varnothing$ *faire*: $A \leftarrow Ct(A)$
*Fin tant que*
- retourne(A)
*Fin* ■

**Propriété 1*.** L'opérateur $Cont_{S1}$ est idempotent, c.-à-d. que $Cont_{S1}(Cont_{S1}(S)) = Cont_{S1}(S)$. ■

**Théorème 1*.** Soit $M_1$ un système à contrôler spécifié par $S_1$, et soit un service désiré spécifié par $S_0$.
Alors, L'AEF $Cont_{S1}(S_0)$ est la spécification $S_c$ du CCPG, telle que $S_c \leq S_1 \times S_0$. ■

**Propriété 2*.** Du théorème 1, nous déduisons : $S_c$ est commandable $\Leftrightarrow$ $S_c = Cont_{S1}(S_c)$. ■

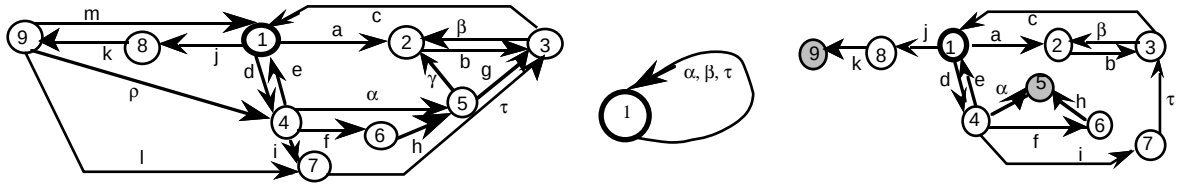
**Lemme 1*.** La complexité de calcul de $S_c$ est d'ordre $n_1^2 \times n_0^2$, (voir Table 1 pour $n_1$ et $n_0$). ■

### 4.2. Calcul des spécifications du contrôleur et du service fourni

La spécification $(S_2, \Psi)$ du contrôleur (déf.7) est trivialement déterminée à partir du CCPG $S_c$. En effet, comme le contrôleur observe tous les événements ($V_{no} = \varnothing$), il n'y a alors aucune projection à faire et $S_2 = S_c = (Q_c, V, \delta_c, q_{c0})$. La fonction $\Psi$ est telle que pour tout état q de $S_2$, $\Psi(q)$ contient tous les événements non commandables et tous les événements tirables à partir de q. Formellement : $\forall\ q \in Q_c : \Psi(q) = V_{nc} \cup \{\sigma \mid \sigma \in V_{cd}$ et $\delta_c(q, \sigma)!\}$. Nous verrons que $\Psi$ est moins triviale dans le cas des systèmes partiellement observables (sect.5). Le service fourni est alors spécifié par $P_0(S_c)$, où $P_0$ la projection sur $V_0$ (Table 1). On peut vérifier que $P_0(S_c) \leq S_0$. En effet, $S_c \leq S_1 \times S_0$, donc $P_0(S_c) \leq P_0(S_1 \times S_0) \leq P_0(S_0) = S_0$.

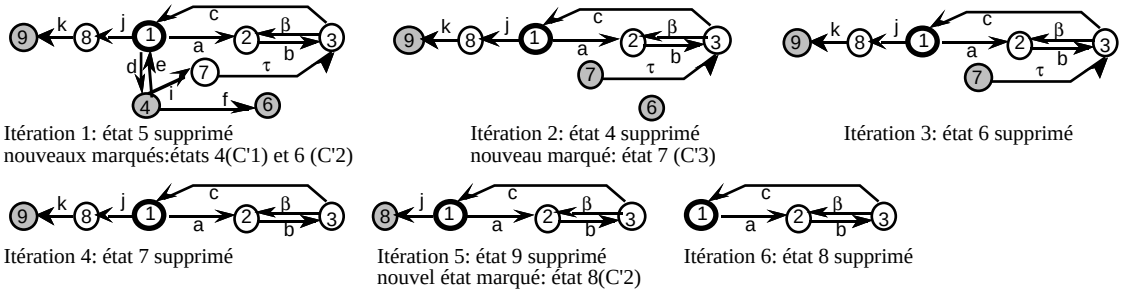
* Preuve en annexe

### 4.3. Exemple




**8.a:** $S_1$ **8.b:** $S_0$ **8.c:** $C_{S1}(S_0)$

**Figure 8.** Exemple de service non commandable



Itération 1: état 5 supprimé
nouveaux marqués:états 4(C'1) et 6 (C'2)

Itération 2: état 4 supprimé
nouveau marqué: état 7 (C'3)

Itération 3: état 6 supprimé

Itération 4: état 7 supprimé

Itération 5: état 9 supprimé
nouvel état marqué: état 8(C'2)

Itération 6: état 8 supprimé

**Figure 9.** Exemple de calcul de $Cont_{S1}(S_0)$

Soit $S_1$ représenté sur la figure 8.a de vocabulaire $V_1=\{a,b,c,d,e,f,g,h,i,j,k,l,m,\alpha,\beta,\gamma,\rho,\tau\}$ avec $V_{nc}=\{\alpha,\beta,\gamma,\rho,\tau\}$. Soit $S_0$ représenté sur la figure 8.b de vocabulaire $V_0=V_{nc}=\{\alpha,\beta,\gamma,\rho,\tau\}$. $S_0$ spécifie que les événements non commandables $\gamma$ et $\rho$ sont interdits. En appliquant $C_{S1}$ à $S_0$ nous obtenons $C_{S1}(S_0)$ représenté sur la figure 8.c ($Q_m$={états 5 et 9} car l'état 5 interdit $\gamma$ et l'état 9 interdit $\rho$). En appliquant l'opérateur Ct sur six itérations (figure 9) nous obtenons $S_c$ (itération 6 de la figure 9). Nous avons alors $S_2=S_c$, et $\Psi$ est telle que $\Psi(1)=V_{nc}\cup\{a\}$, $\Psi(2)=V_{nc}\cup\{b\}$ et $\Psi(3)=V_{nc}\cup\{c\}$. Le service fourni est enfin spécifié par $P_0(S_c)=S_c$, car dans notre exemple $V_0=V_1=V_2$.

### 4.4. Comparaison avec l'algorithme de Ramadge et Wonham

Dans cette section, notre algorithme est défini par l'opérateur $Cont_{S1}$, et celui de Ramadge et Wonham([14]) est désigné par $Alg_{RW}$. Signalons tout d'abord que $Cont_{S1}$ et $Alg_{RW}$ sont de même complexité et utilisent tous les deux la méthode du point fixe. Il y a néanmoins quelques différences que nous allons examiner dans la suite. Soit donc $S_1$ et $S_0$ les spécifications du système à contrôler et du service, de vocabulaires $V_1$ et $V_0$.
(1) $Alg_{RW}$ suppose que $V_1=V_0$ et $S_0\leq S_1$. Pour le généraliser il suffit de prendre $S_{p1}=Comp_V(S_1)$ et $S_{cd}=S_1\times S_0$, qui sont de même vocabulaire $V=V_1\cup V_0$. Dorénavant, nous considérons que les entrées de $Alg_{RW}$ sont $S_{p1}$ et $S_{cd}$ .
(2) $Cont_{S1}$ suppose tous les états de $S_1$ et $S_0$ sont acceptants, alors que $Alg_{RW}$ considère le cas où les états de $S_{cd}$ ne sont pas tous acceptants. $Cont_{S1}$ peut être adapté pour considérer ce cas.
(3) $Alg_{RW}$ n'élimine pas les états absorbants, mais il peut être adapté pour le faire.
(4) $Alg_{RW}$ suppose que tous les états de $S_{p1}$ et $S_{cd}$ sont accessibles à partir de l'état initial.

Nous allons d'une manière non formelle rappeler le principe de $Cont_{S1}$ et décrire le principe de $Alg_{RW}$.

**$Cont_{S1}$** : l'opérateur $C_{S1}$, est appliqué une fois sur $S_0$ pour générer $C_{S1}(S_0)\leq S_1\times S_0=S_{cd}$, dont tous les états sont accessibles à partir de l'état initial, et dont tout état qui interdit un événement tirable non commandable ou qui est absorbant dans $S_1\times S_0$ est alors marqué et blocant dans $C_{S1}(S_0)$ (fig.7 et 8.c). L'opérateur Ct est ensuite utilisé itérativement comme suit. Ct(A) supprime un des états marqués d'un AEFM A. La suppression de celui-ci peut engendrer des états absorbants, des états non accessibles à partir l'état initial, ou des états interdisant des événements tirables non commandables. De tels états sont à leur tour marqués. Soit donc $S_{c,0}=C_{S1}(S_0)$, et $S_{c,i+1}=Ct(S_{c,i})$. Ct est appliqué itérativement jusqu'à obtenir $S_{c,i+1}$ sans états marqués (fig.9), qui spécifie alors le CCPG.

**$Alg_{RW}$ :** On effectue $S_{p1}=Comp_V(S_1)$ et $S_{cd}=S_1\times S_0\leq S_{p1}$ (cette étape n'existe en fait pas dans l'algorithme $Alg_{RW}$ dont les entrées sont directement $S_{p1}$ et $S_{cd}$). On définit l'application unique h qui associe tout état x de $S_{cd}$ à un état q de $S_{p1}$ et telle que x et q sont accessibles par une même séquence d'événements à partir des états initiaux de $S_{p1}$ et $S_{cd}$. On définit l'opérateur $\Omega$ tel que si $B=\Omega(A)$ avec $A\leq S_{cd}$, B est obtenu en : a) supprimant de A tout état x interdisant un événement non commandable tirable de l'état h(x) de $S_{p1}$ ; b)supprimant ensuite tous les états non accessibles générés.
Soit $S_{c,0}=S_{cd}$, et $S_{c,i+1}=\Omega(S_{c,i})$. $\Omega$ est appliqué itérativement jusqu'à ce que $S_{c,i+1}=\Omega(S_{c,i})=S_{c,i}$, qui spécifie alors le CCPG. Si nous reprenons le même exemple (fig.8) que nous avons traité avec notre algorithme nous obtenons: $S_{p1}=S_1$ (car $K_4$ est vide), et $S_{cd}$ est représenté sur la figure 10. Au bout de trois itérations (fig.11), nous obtenons $S_c=S_{c,3}=S_{c,2}$ .

- $Cont_{S1}$ s'est exécuté en 6 itérations alors que $Alg_{RW}$ seulement en 3 itérations. Mais il ne faut pas oublier que dans une itération $Cont_{S1}$ considère un seul état marqué et les différents états qui lui sont directement reliés et qui ne sont pas marqués. Alors que $Alg_{RW}$ considère à chaque itération tous les états. La complexité des deux algorithmes dans le pire cas est d'ordre $n_1{}^2 \times n_0{}^2$, (Table 1 pour $n_1$ et $n_0$). Mais selon les exemples traités, un algorithme peut être plus optimal que l'autre. $Cont_{S1}$ est le plus optimal si le nombre d'états marqués à chaque itération est faible devant le nombre total des états. $Alg_{RW}$ est le plus optimal, si le nombre des états marqués à chaque itération est très important.

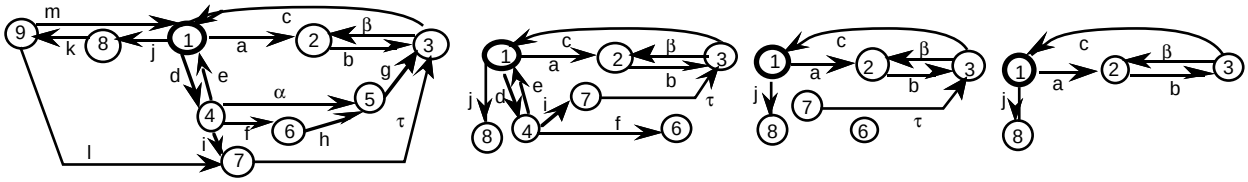



**Figure 10.** Exemple de $S_{cd}=S_1 \times S_0$




Itération 1: $S_{c,1}$
suppression de 9 car ρ interdit
suppression de 5 car γ interdit
il n'y pas d'état non accessible.

Itération 2:
suppression de 4 car α interdit
états 6 et 7 non accessibles.

Itération 2: $S_{c,2}$
après suppression des
états non accessibles.

**Figure 11.** Exemple d'utilisation de $Alg_{RW}$

## 5. Contrôle de systèmes discrets partiellement observables

### 5.1. P-commandabilité et observabilité d'un comportement

Considérons le cas général où certains événements de $M_1$ ne sont pas observables par $M_2$. Nous rappelons que dans ce cas (fig.1) $K_1=V_{no} \cap V_{nc}$, $K_2 \cup K_3 \cup K_4 = V_{ob} \cup V_{cd}$, et $K_4 \subseteq V_{ob} \cap V_{cd}$. Contrairement à l'hypothèse implicite faite par Merlin et Bochmann ([7]), certains événements de $K_2 \cup K_3$ peuvent être non observables (tout en étant commandables). Dans la section 4, comme tous les événements sont observables, un comportement commandable est toujours réalisable. Du fait de la non observabilité de certains événements de $K_2 \cup K_3$ et de tous les événements de $K_1$,un comportement commandable $S_c$ obtenu par l'opérateur $Cont_{S1}$ n'est pas toujours réalisable, même s'il est compatible avec la dynamique du système spécifié par $S_1$. Nous sommes donc amenés à définir la P-commandabilité ([6,11]).

**Définition 13.** (*Comportement P-commandable*)
Soit un système $M_1$ à contrôler spécifié par $S_1$. Un comportement commandable $S_{pc}$ est **P-commandable** s'il peut être atteint, à l'aide d'un contrôleur $M_2$ à déterminer spécifié par $(S_2, \Psi)$, malgré l'observabilité seulement partielle des événements du système. Plus formellement : $V_{no} \neq \emptyset$ et $\exists\ (S_2, \Psi)$, tel que $M_1 \| M_2$ est spécifié par $S_{pc}$. ■

**Remarque :** Lorsque $V_{cd}$ n'est pas inclus dans $V_{ob}$, $M_1 \| M_2$ n'est pas toujours spécifié par $S_1 \times S_2$. En effet, le produit synchronisé implique que $M_2$ (déf.7) ne peut pas interdire des événements n'appartenant pas à $V_{ob}$.

**Définition 14.** (*Comportement observable*)
L'observabilité est définie dans ([6,11]) pour $K_3=K_4=\emptyset$ (fig. 1), nous la redéfinissons dans le cas général. Soit $V=V_1 \cup V_2$, $S_1=(Q_1, V_1, \delta_1, q_{10})$ et $Comp_V(S_1)=(Q_1, V_1 \cup V_2, \delta_{V1}, q_{10})$.Un comportement spécifié par un AEF $S=(Q, V_1 \cup V_2, \delta, q_0)$ plus petit que $Comp_{V(S1)}$, est observable si et seulement si :
$\forall\ s,t \in (V_1 \cup V_2)^*$, $(\delta(q_0,s)! \wedge \delta(q_0,t)!) \Rightarrow$ (P4 $\Rightarrow$ P5). avec : P4=($P_{ob}(s)=P_{ob}(t)$), et
P5= ($\forall\ \sigma \in V_1 \cup V_2$: $(\delta_{V_1}(q_{10},s\sigma)! \wedge \delta_{V_1}(q_{10},t\sigma)!) \Rightarrow (\delta(q_0,s\sigma)! \Leftrightarrow \delta(q_0,t\sigma)!)$). $P_{ob}$ représente la projection sur $V_{ob}$ (Table 1).
D'une manière moins formelle: après une séquence d'événements, les événements permis par le contrôleur $M_2$ ne dépendent que de la projection de cette séquence sur le vocabulaire observable $V_{ob}$. Si nous prenons $K_4=\emptyset$, c.-à-d. $V_2 \subseteq V_1$ et donc $V=V_1$, nous obtenons $Comp_V(S_1)=S_1$ et nous retrouvons une définition analogue à celle de [6,11]. ■

**Théorème 2*.** Un AEF S fermé et plus petit que $Comp_V(S_1)$ est commandable et observable si et seulement si il est P-commandable (déf.13). Une preuve est donnée dans [6] dans le cas où $K_4=K_3=\emptyset$. ■

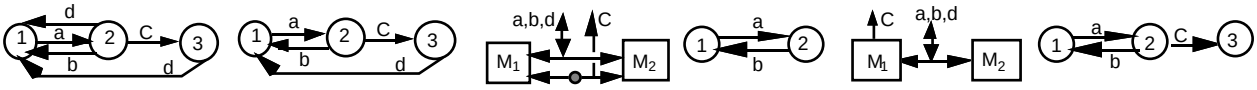
### 5.2. Formule de Merlin-Bochmann dans le cas des comportements P-commandables

Dorénavant, un service $S_0$ est dit P-commandable si le comportement désiré $S_{cd}=S_0 \times S_1$ est P-commandable. La formule de base de Merlin-Bochmann ([7]) est la suivante. Soit un système $M_1$ spécifié par $S_1$ de vocabulaire $V_1$ , et soit un service désiré $S_0$ de vocabulaire $V_0$. Lorsque $V_{ob}=V_{cd}$, alors le contrôleur $M_2$, de vocabulaire $V_2=V_{ob}=V_{cd}$, à joindre à $M_1$ pour satisfaire le service $S_0$, est spécifié par $S_2=P_2(S_0 \times S_1)-P_2(\underline{S_0} \times S_1)$. Et $M_1 \| M_2$ est alors $S_1 \times S_2$ (car $V_{ob}=V_{cd}$). Pour un comportement $S_0 \times S_1$ P-commandable, la formule Merlin-Bochmann se réduit à $S_2=P_2(S_0 \times S_1)$. En effet, $S_0 \times S_1$ P-commandable implique $S=S_1 \times S_0=S_1 \times S_2=S_1 \otimes S_2$. Ceci entraîne $S=S_1 \times P_2(S)=P_1(S) \times P_2(S)$.

* Preuve en annexe

### 5.3. Exemple d'un comportement commandable et non observable

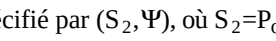
Sur l'exemple de la figure 12, $V_0=V_1=\{a,b,d,C\}$ et seul l'événement C n'est pas observable, c.-à-d. $V_{ob}=\{a,b,d\}$ et $V_{no}=\{C\}$. Les actions a,b et d sont toujours commandables. Le comportement $S_0\times S_1$ est commandable, mais il n'est pas P-commandable car il n'est pas observable. En effet, d'une manière informelle, C étant non observable, $M_2$ ne peut pas savoir si on est dans l'état 2 ou 3, et ne peut donc pas décider d'interdire l'action d. Considérons les deux cas suivants.
*Cas 1: C est commandable* (fig.13.a) : Le comportement $S_{pc}$ observable et commandable (donc P-commandable) le plus grand tel que $S_{pc}\leq S_0\times S_1$, est spécifié par l'AEF de la figure 13.b . Comme $V_2=V_1=V_0$, le service fourni est $S_0=S_{pc}$.
*Cas 2: C n'est pas commandable* (fig.14.a) : La formule générale de Merlin-Bochmann peut être utilisée (car $V_{ob}=V_{cd}$, et donc $V_2=V_{ob}=V_{cd}$) afin d'extraire un comportement observable. Soit donc $S_2=P_2(S_0\times S_1)$-$P_2(\underline{S_0}\times S_1)$. Et $S=S_1\times S_2$ (fig.14.b) est le comportement observable le plus grand qui soit plus petit que $S_0\times S_1$. Mais S n'est pas commandable, car il possède un état blocant (déf.1). Le comportement P-commandable le plus grand qui soit plus petit que $S_0\times S_1$ est alors spécifié par un seul état sans l'occurrence d'aucun événement.




$S_1$ $S_0=S_1\otimes S_0=S_1\times S_0$ **13a.** **13.b.** **14.a.** **14.b.**

**Figure 12.** $S_1\times S_0$ commandable et non observable **Figure 13.** Cas 1 **Figure 14.** Cas 2

### 5.4. Exemple de comportement commandable et observable

Sur l'exemple de la figure 15, $V_1=\{a,b,d,e,C,F\}$, $V_0=\{a,d,e,C,F\}$, $V_2=\{a,b,d,e,F\}$, $V_{cd}=\{a,b,d,e,F\}$, $V_{ob}=\{a,b,d,e\}$, et donc $V_{nc}=\{C\}$ et $V_{no}=\{C,F\}$. Le comportement $S_{pc}=S_0\times S_1$ peut être réalisé, car celui-ci est observable et commandable, et donc P-commandable. Celui-ci peut être obtenu si le contrôleur $M_2$ interdit les mêmes événements, e et F, avant et après l'événement non observable C. $M_2$ est alors spécifié par $(S_2,\Psi)$, où $S_2=P_{ob}(S_{pc})=$ 

, et $\Psi$ est telle que $\Psi(1)=V_{nc}\cup\{a\}$ et $\Psi(2)=V_{nc}\cup\{b,d,C\}$. Nous voyons ici l'utilité de $\Psi$. En effet, $\Psi(q)$ ne contient pas forcément tous les événements non observables. Ce qui signifie que le contrôleur peut en effet interdire des événements commandables même s'il ne peut pas les observer (événement F). Ceci explique pourquoi $M_1\|M_2$ n'est pas forcément spécifié par $S_1\times S_2$ , si $V_{cd}$ n'est pas inclus dans $V_{ob}$. Le service obtenu est $P_0(S_{pc})=S_0$ .


$S_1$ $S_0$ $S_0\times S_1$

**Figure 15.** Comportement commandable et observable

### 5.5. Comportement T-observable

Nous avons donné précédemment un exemple d'extraction du comportement $S_{pc}$ P-commandable le plus grand tel que $S_{pc}\leq S_0\times S_1$. Mais dans le cas général, un tel comportement n'existe pas toujours ([6,11]), car l'union de plusieurs comportements observables n'est pas toujours observable. De plus, les résolutions de problèmes avec des informations partiellement observables sont NP-complets ou pire ([8,9,12]). Pour ces raisons, une classe de comportements observables est définie. Les comportements de cette classe seront dits T-observables (T pour totalement).

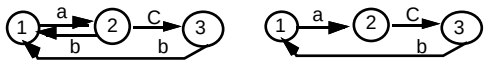
**Définition 15.** (*Comportement T-observable*)
Si le système à contrôler est spécifié par $S_1$, un comportement spécifié par S est dit T-observable si et seulement si: $S=S_1\times P_{ob}(S)$ , où $P_{ob}$ est la projection sur le vocabulaire (alphabet) observable $V_{ob}$. ■

Un tel comportement, qualifié de Normal dans [6,11], a des propriétés intéressantes qui sont :
- l'union de plusieurs comportements T-observables est T-observable. Il existe donc un plus grand comportement T-observable (CTPG) $S_{to}$ tel que $S_{to}\leq S_1\times S_0$.
- si S est T-observable alors il dépend uniquement de sa projection dans $V_{ob}$. C'est en fait le comportement le plus grand tel que $S\leq Comp_V(S_1)$ et ayant $P_{ob}(S)$ comme projection.
- pour toute séquence s respectant $S_1$, nous pouvons décider si elle appartient à S à partir de sa projection $P_{ob}(s)$.
- si S est T-observable alors il est observable ([6,11]), mais la réciproque n'est pas vraie. Donc **si S est fermé, commandable et T-observable alors il est P-commandable.** Une autre propriété que nous ajoutons est la suivante.

**Propriété 3**[*]**.** Si S est T-observable alors les événements non observables de $V_1$ ne sont pas interdits. Autrement dit, les événements non observables sont considérés comme non commandables, c.-à-d. $V_{cd}\subseteq V_{ob}$. Dans ce cas, nous avons alors $V_2=V_{ob}$. ■

[*] Preuve en annexe

La réciproque de la dernière propriété est fausse. Voici un contre-exemple avec $V_{ob}=\{a,b\}$. On a $S_1\times S_0=S_0$ qui consiste à interdire l'événement observable b à l'état 2. Mais on peut vérifier facilement que $S\leq S_1\times P_{ob}(S)=S_1$.



$S_1$ S

**Remarque:** on peut penser que la contrainte de considérer que les comportements non observables sont non commandables est trop forte. Mais on montre dans [6] que de tels comportement ont plusieurs propriétés intéressantes, et se retrouvent souvent dans des applications réelles.

**Théorème 3*.** Soit le système $M_1$ à contrôler spécifié par $S_1$ et soit le service $S_0$ de vocabulaire $V_0$. Si $S_1\times S_0$ n'est pas T-observable, on peut déterminer le comportement $S_{to}$ T-observable le plus grand (CTPG) tel que $S_{to}\leq S_1\times S_0$, par un algorithme résolvant la formule de Merlin-Bochmann ([7]). $S_{to}$ est alors : $S_{to}=S_1\times S_2$, avec $S_2=P_2(S_0\times S_1)\text{-}P_2(\underline{S}_0\times S_1)$, où $P_2$ représente la projection sur $V_{ob}$, car $V_2=V_{ob}$ (prop.3). Plus particulièrement, si $V_0=V$, et si $S_0\leq Comp_V(S_1)$, alors le CTPG $S_{to}$ est donné par : $S_{to}=S_1\times S_2$, avec $S_2=P_2(S_0)\text{-}P_2(\underline{S}_0\times S_1)$.
Signalons que $S_{to}$ peut contenir des états blocants. ■

**Définition 16.** (*Opérateur $Tob_{S1}$*)
On définit maintenant l'opérateur $Tob_{S1}$ qui permet d'obtenir le CTPG $S_{to}$ tel que $S_{to}\leq S_1\times S_0$, c.-à-d. $S_{to}=Tob_{S1}(S_0)=S_1\times(P_2(S_0\times S_1)\text{-}P_2(\underline{S}_0\times S_1))$ avec, rappelons le, $V_2=V_{ob}$. Autrement dit, la T-observabilité peut être définie par : $S_{to}$ est T-observable $\Leftrightarrow S_{to}=Tob_{S1}(S_{to})$.

Si $n_0$ et $n_1$ sont les nombres d'états de $S_0$ et $S_1$, alors $Tob_{S1}(S_0)$ est obtenu comme suit.

**1) Obtention de $\underline{S}_0$.** Pour représenter tout d'abord $\underline{S}_0$ nous devons définir l'automate $S_p$ suivant. S et soit $S_p$ l'automate à $n_0+1$ états tels que:
$*$ $n_0$ états correspondent aux $n_0$ états de $S_0$
$*$ le $(n_0+1)$ième état est marqué et est atteint par toute séquence d'événements n'appartenant pas à $S_0$.
$S_p$ est défini formellement par: si $S_0=(Q_0,V_0,\delta_0,q_{00})$ alors $S_p=(Q_p,V_0,\delta_p,q_{00},\{q_m\})$ tel que $Q_p=Q_0\cup\{q_m\}$, où $q_m$ est l'état marqué de $S_p$ et :

(a) $\forall\, q\in Q_p,\ \forall\, \sigma\in V_0 :\ \delta_p(q,\sigma)!$
(b) $\forall\, q\in Q_0,\ \forall\, \sigma\in V_0 :\ \delta_0(q,\sigma)!\ \Rightarrow\ \delta_p(q,\sigma)=\delta_0(q,\sigma)\neq q_m$
(c) $\forall\, q\in Q_0,\ \forall\, \sigma\in V_0 :\ \delta_0(q,\sigma)\neg!\ \Rightarrow\ \delta_p(q,\sigma)=q_m$
(d) $\forall\, \sigma\in V_0 :\ \delta_p(q_m,\sigma)=q_m$

(a) signifie que $S_p$ contient toutes les séquences de $V_0^*$ ,
(b) signifie que les séquences de $S_0$ ne mènent pas à l'état marqué dans $S_p$,
(c) signifie que les séquences de $S_p$ n'appartenant pas à $S_0$ mènent à l'état marqué,
(d) signifie que l'état marqué est absorbant.
Nous pouvons alors déduire que les séquences menant à $q_m$ sont celles qui appartiennent à $\underline{S}_0$, et la complexité d'obtention de $S_p$ est bornée par $|V_0|\times n_0$.

**2) Obtention de $\underline{S}_0\times S_1$.** Si $S_1=(Q_1,V_1,\delta,q_{10})$ avec $V=V_1\cup V_0$, le produit synchronisé et marqué $S_{p1}=S_p\times S_1$ est défini par $S_{p1}=(Q_{p1},V,\delta_{p1},q_{p0},Q_{pm})$ avec : $(Q_{pm}\subseteq\{q_m\}\times Q_1)$ et $(\forall\ q\in Q_{p1} : q=\langle q_m,q_1\rangle\Rightarrow\ q\in Q_{pm})$. Nous déduisons que les séquences menant aux états marqués de $S_{p1}$ sont celles qui sont acceptées par $\underline{S}_0\times S_1$, et la complexité d'obtention de $S_{p1}$ est bornée par $|V_0|\times n_0\times n_1$.

**3) Obtention de $P_2(\underline{S}_0\times S_1)$.** Soit alors la projection $P_2(S_{p1})$, ayant un ensemble d'états marqués. Les séquences de $P_2(S_{p1})$ menant aux états marqués sont celles qui appartiennent à $P_2(\underline{S}_0\times S_1)$ . .

**4) Obtention de $P_2(S_0\times S_1)\text{-}P_2(\underline{S}_0\times S_1)$.** $P_2(S_0\times S_1)\text{-}P_2(\underline{S}_0\times S_1)$ est égal $P_2(S_0\times S_1)\nabla P_2(S_{p1})$ où $\nabla$ signifie produit synchronisé avec élimination des états marqués. Comme les nombres d'états de $P_2(S_0\times S_1)$ et $P_2(S_{p1})$ sont bornés par $n_0\times n_1$, la complexité de $\nabla$ est bornée par $|V_2|\times(n_0\times n_1)^2$.

Des exemples de calcul de $S_2$ sont donnés dans [7].
Une projection $P_2$ de $S_0\times S_1$ ou $S_{p1}$ sur un vocabulaire $V_2$ consiste en trois principales étapes qui sont:

1- remplacer les événements de $V\text{-}V_2$ par la transition $\varepsilon$ : la complexité est en $O(n_0\times n_1\times|V|)$
2- supprimer les cycles de transitions $\varepsilon$
3- supprimer les séquences non cycliques restantes de transitions $\varepsilon$ : la complexité est en $O(n_0\times n_1\times|V|^2)$ Des algorithmes réalisant les étapes deux et trois sont proposés dans [1]. ■

* Preuve en annexe

**Propriété 4*.** L'opérateur $Tob_{S1}$ est idempotent, c.-à-d. que $Tob_{S1}(Tob_{S1}(S))= Tob_{S1}(S)$. ■

**Propriété 5*.** $Tob_{S1}(S_0)=Tob_{S1}(Comp_V(S_0))$. ■

### 5.6 Extraction du comportement T-observable et commandable le plus grand: opérateur $Toc_{S1}$

Le comportement T-observable calculé par l'opérateur $Tob_{S1}$ n'est pas forcément commandable. Il faut donc en extraire un comportement commandable et qui reste T-observable, désigné par CCTPG. Nous définissons alors l'opérateur $Toc_{S1}$ qui permet de calculer un tel comportement.

**Définition 17.** (*Opérateur $Toc_{S1}$*)
A partir de $S_0$ et de $S_1$, l'opérateur $Toc_{S1}$ utilise les deux opérateurs $Cont_{S1}$ et $Tob_{S1}$ comme suit.

**$Toc_{S1}(S_0)$** *Début*: - arrêt ← faux
- $S_{to}$ ← $Tob_{S1}(S_0)$
- *Tant que* ¬ arrêt *faire*
  $S_c$ ← $Cont_{S1}(S_{to})$
  *Si* ($S_c = S_{to}$) *alors* arrêt ← vrai
  *Sinon* - $S_{to}$ ← $Tob_{S1}(S_c)$
  - *Si* ($S_c=S_{to}$) *alors* arrêt ← vrai
- *Fin tant que*
- retourne($S_c$)

*Fin* ■

L'opérateur $Toc_{S1}$ est plus complexe que la procédure proposée dans [3], celle-ci n'étant pas itérative. Mais dans notre cas, le comportement obtenu ne contient pas d'état blocant, ce qui n'est le cas ni dans [3] ni dans [7].

**Théorème 4*.** Soit $M_1$ un système à contrôler spécifié par $S_1$, et soit un service désiré spécifié par $S_0$.
Alors, L'AEF $Toc_{S1}(S_0)$ est la spécification $S_{ct}$ du CCTPG, telle que $S_{ct} \leq S_1 \times S_0$. ■

### 5.7. Calcul des spécifications du contrôleur et du service fourni

Comme $V_{cd} \subseteq V_{ob}=V_2$ (prop.3), alors $(S_2,\Psi)$ est, comme dans le cas $V_{no}=\emptyset$, trivialement déterminée à partir du CCTPG $S_{ct}$. En effet, $S_2=P_2(S_{ct})=P_{ob}(S_{ct})=(Q_2,V_2,\delta_2,q_{20})$, et la fonction $\Psi$ est telle que pour tout état q de $S_2$, $\Psi(q)$ contient tous les événements non commandables et tous les événements tirables à partir de q. Formellement :

$$\forall\ q \in Q_2 : \Psi(q)=V_{nc} \cup \{\sigma \mid \sigma \in V_{cd} \text{ et } \delta_2(q,\sigma)!\}.$$

Le service fourni est tout simplement spécifié par $P_0(S_{ct})$.

Dans le cas général où $V_{cd} \not\subset V_{ob}$, et donc $V_2 \neq V_{ob}$, le contrôleur $(S_2,\Psi)$ permettant de réaliser un comportement P-commandable (déf.13) spécifié par $S_{pc}=(Q_{pc},V,\delta_{pc},q_{pc0})$, est obtenu comme suit. $S_2=P_{ob}(S_{pc})=(Q_2,V_{ob},\delta_2,q_{20}), \neq P_2(S_{pc})$. (car $V_2 \neq V_{ob}$,). Pour déterminer $\Psi$, définissons la fonction $\Phi$ qui associe à tout état q de $S_{pc}$, l'union des ensembles des événements non commandables et des événements commandables tirables à partir de q. Formellement :
$\forall\ q \in Q_{pc} : \Phi(q)=V_{nc} \cup \{\sigma \mid \sigma \in V_{cd} \text{ et } \delta_{pc}(q,\sigma)!\}$. $\Psi$ est alors formellement déterminée à partir de $\Phi$ comme suit.
$\forall\ q \in Q_2 : \Psi(q)=\Phi(q_1) \cup \Phi(q_2) \cup ... \cup \Phi(q_n)$, avec $q_1, q_2, ..., q_n \in F(q)$ et :

$$F(q)=\{q_i \mid (q_i \in Q_{pc}) \wedge (\exists\ s_i \in V^* \mid (\delta_{pc}(q_{pc0},s_i)=q_i) \wedge (\delta_2(q_{20},P_{ob}(s_i))=q))\}.$$

### 6. Conclusion

Dans cet article, nous montrons qu'une étude rigoureuse de la synthèse de contrôleur doit prendre en compte la commandabilité et l'observabilité des événements du système à contrôler. Nous avons ainsi étendu la méthode décrite dans [7] en utilisant la théorie du contrôle des systèmes à événements discrets (SED) et en générant des comportements sans état blocant. Cette théorie considère les ensembles $K_3$ et $K_4$ vides (fig.1). Nous montrons que l'étude du contrôle peut aisément être faite dans le cas plus général où $K_3$ et/ou $K_4$ sont non vides.

Lorsque le système à contrôler est totalement observable, nous proposons un algorithme pour extraire le comportement commandable le plus grand. Cet algorithme est différent de celui décrit dans [14], mais il utilise lui aussi la méthode du point fixe et il est de même complexité dans le pire cas. Mais selon les cas traités, les complexités diffèrent. Nous en déduisons que les deux algorithmes sont complémentaires.

Pour l'étude des systèmes partiellement observables, on est confronté à un problème de complexité. Pour cette raison, une sous-classe de comportements partiellement observables définis dans [6], et que nous avons appelés T-observables, sont étudiés. Nous les avons considérés car ils possèdent plusieurs propriétés intéressantes. Nous montrons qu'ils peuvent être obtenus par un algorithme résolvant la formule (1) donnée dans [7]. Les principales étapes de résolution de cette formule sont passées en revue. Nous proposons ensuite un algorithme de calcul du comportement

* Preuve en annexe

le plus grand qui soit commandable et T-observable et qui satisfait un service désiré.
Contrairement à la procédure décrite dans [3], notre algorithme ne génère pas d'état blocant.
Dans le futur, nous proposons les extensions suivantes :
- la prise en compte des contraintes temporelles entre événements quelconques (c.-à-d non forcément consécutifs). Une étude en ce sens est entamée.
- la répartition du contrôle pour des systèmes distribués concurrents. Là aussi, une étude est en cours.

**7. Bibliographie**


**[1]** W.A. Barrett et J.D. Couch,"Compiler Construction: Theory and Practice." Editeur: Science Research Associates, Inc. 1979.

**[2]** H. Cho et S.I. Marcus, "On the Supremal Languages of Sublanguages that Arise in Supervisory Synthesis Problems with Partial Observation," Mathematics Contr.,Signals Syst., Vol.2, No.2, pp.47-69, 1989.

**[3]** R. Cieslak, C. Desclaux, A. Fawaz et P. Varaiya, "Supervisory Control of Discrete Event Processes with Partial Observations," IEEE Transactions on Automatic Control, Vol.33, No.3, pp.249-260, Mars 1988.

**[4]** M.H. Erdogmus et R. Johnston, "On the Specification and Synthesis of Communicating Processes," IEEE Transactions on Software Engineering, Vol.16, No.12, Décembre 1990.

**[5]** A. Khoumsi, G.v. Bochmann et R. Dssouli, "Contrôle et extension des systèmes à événements discrets totalement et partiellement observables," Third Maghrebian Conference on Software Engineering and Artificial Intelligence, Rabat, April 1994.

**[6]** F.Lin et W.M. Wonham, "On observability of Discrete-Event Systems," Information Sciences, Vol.44, pp.173-198, 1988.

**[7]** P. Merlin et G.v. Bochmann, "On the Construction of Submodule Specifications and Communication Protocols," ACM Transactions on Programming Languages and Systems, Vol.5, No.1, Janvier 83.

**[8]** C.H. Papadimitriou et J.N. Tsitsiklis, "On the complexity of designing distributed protocols," Inform. Contr., Vol.53, pp. 211-218, Juin 1982.

**[9]** C.H. Papadimitriou et J.N. Tsitsiklis, "On the complexity of Markov decision processes," Mathematics Operations Res., Vol. 12, No.3, pp. 441-450, Août 1987.

**[10]** P.J.G. Ramadge et W.M. Wonham, "Modular Feedback Logic For Discrete Event Systems," SIAM J. Contr. Optimization, Vol.25, No.5, pp. 1202-1218, Mai 1987.

**[11]** P.J.G. Ramadge et W.M. Wonham, "The Control of Discrete Event Systems," Proceedings of the IEEE, Vol.77, No.1, Janvier 1989.

**[12]** J.N. Tsitsikis, "On the control of discrete event dynamical systems," Math. Contr., Signals, and Syst., Vol.2, No.1, pp. 95-107, 1989.

**[13]** W.M. Wonham, "A Control Theory for Discrete-Event Systems," In Advanced Computing Concepts and Techniques in Control Engineering, M.J. Denham et A.J. Laub, Eds., Springer-Verlag NATO ASI Series. New York, NY: Springer-Verlag, pp.129-169, 1988.

**[14]** W.M. Wonham et P.J. Ramadge, "On the Supremal Controllable sublanguage of a Given Language," SIAM J.Control and Optimization, Vol.25, No.3, Mai 1987.

# ANNEXE : Preuves

**Preuve.** (Propriété 1)
Par construction (def.12), $Cont_{S1}(S_0)$ est commandable et ne contient donc pas d'état marqué. (1)
Par construction (def.10), si B est commandable alors $C_{S1}(B)=B$. (2)
Par construction (def.11), si B est sans état marqué alors Ct(b)=B. (3)
De (2) et (3), nous déduisons que si B est commandable alors $Cont_{S1}(B)=B$. (4)
De (1) et (4), nous déduisons que $Cont_{S1}(Cont_{S1}(S))= Cont_{S1}(S)$. Donc $Cont_{S1}$ est idempotent. ■

**Preuve.** (Théorème 1)
- Montrons tout d'abord que $S_c$ est commandable.
$S_c$ ne contient pas d'état marqué. Les conditions C1, C2 et C3 de la définition 8 sont donc vérifiées pour tout état de $S_c$. Celui-ci est par conséquent commandable.

- Montrons que tout comportement $S_{pc}$ commandable différent de $S_c$ et plus petit que $S_1 \times S_0$, est aussi plus petit que $S_c$ (c.-à-d. que $S_c$ est le CCPG ).
Comme $S_1 \times S_0 \le Comp_V(S_0)$ (déf.4) et $S_{pc} \le S_1 \times S_0$ impliquent $S_{pc} \le Comp_V(S_0)$ (1)
$Cont_{S1}(S_0)=Ct^n(C_{S1}(S_0))$, avec n tel que $Ct^{n+1}(C_{S1}(S_0))=Ct^n(C_{S1}(S_0))$ (2)
où $Ct^n$ signifie que l'opérateur est appliqué n fois.
Comme $C_{S1}(S_0)=C_{S1}(Comp_V(S_0))$ alors $Cont_{S1}(S_0)=Cont_{S1}(Comp_V(S_0))$ (3)
Si A est commandable et $A \le S_1 \times S_0$, alors $A=C_{S1}(A)=Ct(A)$ et donc $A=Cont_{S1}(A)$ (4)
Soit $A \le B$ et soit $n_A$ et $n_B$ tels que $Ct^{n_A}(C_{S1}(A))$ et $Ct^{n_B}(C_{S1}(B))$ sont commandables. Alors :
$C_{S1}(A) \le C_{S1}(B)$ et $Ct^{n_A}(C_{S1}(A)) \le Ct^{n_A}(C_{S1}(B))$ et donc $Cont_{S1}(A) \le Cont_{S1}(B)$ (5)
Comme $S_{pc}$ est commandable et plus que $S_1 \times S_0$, alors (4) implique $S_{pc}=Cont_{S1}(S_{pc})$ (6)
(2) et (3) impliquent $S_c=Cont_{S1}(Comp_V(S_0))$ (7)
(1) et (5) impliquent $Cont_{S1}(S_{pc}) \le Cont_{S1}(Comp_V(S_0))$ (8)
(6) et (8) impliquent impliquent $S_{pc} \le Cont_{S1}(Comp_V(S_0))$ (9)
(7) et (9) impliquent $S_{pc} \le S_c$ et $S_c$ est donc le CCPG. ■

**Preuve.** (Propriété 2)
Pour démontrer que : $S_c$ est commandable $\Leftrightarrow S_c=Cont_{S1}(S_c)$, nous démontrerons les implications dans les deux sens.
1) Hypothèse : $S_c$ est commandable. Du (4) de la preuve de la propriété 1, nous déduisons $S_c=Cont_{S1}(S_c)$.
2) Hypothèse : $S_c=Cont_{S1}(S_c)$. Du théorème 1, nous déduisons que $S_c$ est commandable. ■

**Preuve.** (Lemme 1)
Soit $n_1$ et $n_0$ respectivement les nombres d'états de $S_1$ et $S_0$. La complexité de calcul de $C_{S_1}(S_0)$ est au plus en $n_1 \times n_0$. A chaque itération de Ct, nous avons considéré un seul état marqué et les états qui lui sont directement reliés "en amont et en aval". La complexité d'une itération est donc bornée par le nombre d'états, càd $n_1 \times n_0$. On a au plus $n_1 \times n_0$ itérations, la complexité de calcul de $S_c$ est donc bornée par $n_1^2 \times n_0^2$. ■

**Preuve.** (Théorème 2)
Dans [6], la preuve du théorème est faite lorsque $K_3$ et $K_4$ sont vides. Nous avons vu dans la section 3.2 que le problème du contrôle de $S_1$ pour un service désiré $S_0$ est équivalent au problème du contrôle de $S_{p1}=Comp_V(S_1)$ pour un comportement désiré $S_{p0}=Comp_V(S_0)$.
Comme $S_{p1}$ et $S_{p0}$ ont un même vocabulaire alors $K_3$ et $K_4$ sont vides, et la preuve de [4] est valable. ■

**Preuve.** (Propriété 3)
Soit un AEF S tel que $S=S_1 \times P_{ob}(S)$, (où le produit de $S_1$ et $P_{ob}(S)$ est synchronisé sur $V_{ob} \cap V_1$). Soit s une séquence d'événements telle que $s \in S$ (déf.3), et soit $\sigma$ un événement non observable tel que $s\sigma \in Comp_V(S_1)$, où $V=V_1 \cup V_2$. Comme $P_{ob}(s\sigma)=P_{ob}(s) \in P_{ob}(S)$, et $s\sigma \in Comp_V(S_1)$, alors $s\sigma \in S_1 \times P_{ob}(S)$. D'où si $s \in S$ et $s\sigma \in Comp_V(S_1)$ avec $\sigma \in V_{no}$, alors $s\sigma \in S$. Les événements non observables ne sont donc jamais interdits, et donc $V_{cd} \subseteq V_{ob}$.
Comme $V_2=V_{ob} \cup V_{cd}$ et $V_{cd} \subseteq V_{ob}$, alors $V_2=V_{ob}$. ■

**Preuve.** (Théorème 3)
- Montrons tout d'abord que $S_{to}=S_1 \times S_2$, est T-observable, c.-à-d. que $S_{to}=S_1 \times P_{ob}(S_{to})$. On démontre cette égalité par les inégalités dans les deux sens

1) $S_{to} \le S_1 \times P_{ob}(S_{to})$ :
Soient A et B deux AEFs définis sur des alphabets $V_A$ et $V_B$. Alors $A \times B=A \times Comp_{V_A \cup V_B}(B)$ (1)
Soit C un AEF défini sur un alphabet $V_C$., et soit $V_D$ tel que $V_D \subseteq V_C$. Alors $C \le Comp_{V_C}(P_D(C))$ (2)

$S_{to}=S_1\times S_2=S_1\times(S_1\times S_2)$ (3)
De (2), nous déduisons $S_1\times S_2\leq Comp_V(P_{ob}(S_1\times S_2))$ (4)
De (3) et (4), nous déduisons $S_{to}\leq S_1\times Comp_V(P_{ob}(S_1\times S_2))$ (5)
De (1) et (5), nous déduisons $S_{to}\leq S_1\times P_{ob}(S_1\times S_2)$ (6)
Et enfin, de (3) et (6), nous concluons $S_{to}\leq S_1\times P_{ob}(S_{to})$.

2) $S_1\times P_{ob}(S_{to})\leq S_{to}$:
$S_1\times P_{ob}(S_{to}) = S_1\times P_{ob}(S_1\times S_2)\leq S_1\times P_{ob}(S_2)= S_1\times S_2= S_{to}$.

Des deux inégalités , nous déduisons donc $S_{to}=S_1\times P_{ob}(S_{to})$.

- Montrons maintenant que $S_{to}=S_1\times S_2$ est le CTPG tel que $S_{to}\leq S_1\times S_0$.
De [7] nous déduisons que $S_2$ est le contrôleur le plus grand tel que $P_0(S_1\times S_2)\leq P_0(S_1\times S_0)$ et tel que toute séquence de $S_2$ est compatible avec $S_1$ (conditions 2 et 3 de [7]). Or :
P1 : En plus de $P_0(S_1\times S_2)\leq P_0(S_1\times S_0)$, nous avons aussi $S_1\times S_2\leq S_1\times S_0$.
P2 :Si $S_{to}$ T-observable ($S_{to}=S_1\times P_{ob}(S_{to})$) alors toute séquence de $P_{ob}(S_{to})$ est compatible avec $S_1$
On déduit de P1 et P2 que $S_{to}=S_1\times S_2$ est le CTPG tel que $S_{to}\leq S_1\times S_0$.

*Preuve* de P1: Nous allons montrer que si une séquence $t\in S_1\times S_2$ alors $t\in S_1\times S_0$.
Soit $t\in S_1\times S_2$, donc $t\in Comp_V(S_1)$ et $t\in Comp_V(P_2(S_0\times S_1)\text{-}P_2(\underline{S}_0\times S_1))$ (1)
$t\in Comp_V(S_1)\Rightarrow \exists\,\alpha\in S_1$ tel que $\alpha=P_1(t)$ (2)
$t\in Comp_V(P_2(S_0\times S_1)\text{-}P_2(\underline{S}_0\times S_1))\Rightarrow \exists\,\beta\in P_2(S_0\times S_1)$ tel que :$(\beta\notin P_2(\underline{S}_0\times S_1)\wedge\beta=P_2(t))$ (3)
$\beta\in P_2(S_0\times S_1)\Rightarrow \exists\,\lambda\in S_0\times S_1$ tel que $\beta=P_2(\lambda)$ (4)
$\beta\notin P_2(\underline{S}_0\times S_1)\Rightarrow \forall\,\delta\in\underline{S}_0\times S_1 : \beta\neq P_2(\delta)$ (5)
(3) et (5) $\Rightarrow \forall\,\delta\in\underline{S}_0\times S_1 : P_2(t)\neq P_2(\delta)$ et donc $t\neq\delta$ (6)
(6) $\Rightarrow t\notin\underline{S}_0\times S_1$, c.-à-d. $(t\notin Comp_V(\underline{S}_0))\vee(t\notin Comp_V(S_1))$ (7)
(7) et (1) $\Rightarrow t\in Comp_V(S_1)$ et $t\notin Comp_V(\underline{S}_0)$, et donc $t\in S_1\times S_0$

*Preuve* de P2:
$S_{to}=S_1\times P_{ob}(S_{to})\Rightarrow\{\forall\, s_2\in P_{ob}(S_{to}), \exists\, s\in S_1\times P_{ob}(S_{to})$ telle que $s_2=P_{ob}(s)\}$ (1)
$s\in S_1\times P_{ob}(S_{to})\Rightarrow P_1(s)\in S_1$ (2)
(1) et (2) impliquent que $P_{ob}(S_{to})$ est compatible avec $S_1$ . ■

**Preuve.** (Propriété 4)
Soit $S_{to}=Tob_{S1}(S_0)$ et soit $S=Tob_{S1}(S_{to})$. Montrons que $S=S_{to}$.
$S_{to}$ est donc le CTPG tel que $S_{to}\leq S_1\times S_0$ (déf.16). (1)
S est donc le CTPG tel que $S\leq S_1\times S_{to}$ (déf.16). (2)
De (1) nous déduisons $S_1\times S_{to}=S_{to}$. (3)
De (2) et (3), nous déduisons que S est le CTPG tel que $S\leq S_{to}$ . (4)
Comme $S_{to}$ est T-observable, nous déduisons de (4) que $S=S_{to}$ .

**Preuve.** (Propriété 5)
Comme $Tob_{S1}(S_0)=S_1\times(P_2(S_0\times S_1)\text{-}P_2(\underline{S}_0\times S_1))$, alors pour montrer que $Tob_{S1}(S_0)=Tob_{S1}(Comp_V(S_0))$, il suffit donc de montrer que :$S_0\times S_1=Comp_V(S_0)\times S_1$ et que $\underline{S}_0\times S_1=\underline{Comp}_V(S_0)\times S_1$ .
La première égalité est triviale. Pour la seconde, il suffit de montrer que $\underline{Comp}_V(S_0))=Comp_V(S_0))$. ■

**Preuve.** (Théorème 4)
- Montrons tout d'abord que $S_\alpha=Toc_{S1}(S_0)$ est commandable et T-observable.
Sachant que $Cont_{S1}$ et $Tob_{S1}$ sont idempotents (prop.1 et 4), il est facile de vérifier que $S_\alpha=Cont_{S1}(S_\alpha)=Tob_{S1}(S_\alpha)$.
$S_\alpha$ est donc commandable et T-observable.

- Montrons que tout comportement $S_p$ commandable et T-observable tel que $S_p\leq S_1\times S_0$ vérifie alors $S_p\leq S_\alpha$.
$S_p\leq S_1\times S_0$ alors $S_p\leq Comp_V(S_0)$ (1)
$Toc_{S1}$ est schématisé par $S_\alpha=Toc_{S1}(S_0)=...(Cont_{S1}(Tob_{S1}(...Cont_{S1}(Tob_{S1}(S_0))...)$ (2)
$Toc_{S1}(S_0)=Toc_{S1}(Comp_V(S_0))$ (car $Tob_{S1}(S_0)=Tob_{S1}(Comp_V(S_0))$) (3)
Si A commandable et T-observable et $A\leq S_1\times S_0$, alors : $A=Cont_{S1}(A)=Tob_{S1}(A)=Toc_{S1}(A)$ (4)
Si $A\leq B$ alors $Cont_{S1}(A)\leq Cont_{S1}(B)$ et $Tob_{S1}(A)\leq Tob_{S1}(B)$ et donc $Toc_{S1}(A)\leq Toc_{S1}(B)$ (5)
Soit alors $S_p$ commandable et T-observable et plus petit que $S_1\times S_0$ :
(4) implique $S_p=Toc_{S1}(S_p)$ (6)
(2) et (3) impliquent $S_\alpha=Toc_{S1}(Comp_V(S_0))$ (7)
(1) (6) (7) et (5) impliquent : $S_p\leq S_\alpha$ ■