

Du réseau de Petri au Grafcet

Robert Valette

LAAS-CNRS, F-31077 Toulouse Cedex 4
version provisoire du 7 septembre 2001

1 Historique

Le Grafcet a été initialement défini par une commission d'une société savante aujourd'hui disparue : l'AFCECET (Association Française de Cybernétique Economique et Technique). Le document final de la commission date de 1977.

Au sein de cette commission il y avait des universitaires qui travaillaient sur les réseaux de Petri et l'on peut donc considérer que les réseaux de Petri sont l'un des ancêtres du Grafcet.

Aux réseaux de Petri, il fallait associer une façon normalisée de définir comment le système de commande interagissait avec l'environnement au moyen de capteurs et d'actionneurs. Il fallait donc une norme pour l'interprétation (dans le sens donné dans le cours photocopié [1]) des réseaux de Petri). C'est ce qui a donné les réceptivités associées aux transitions et les actions associées aux étapes dans le Grafcet.

Toutefois, entre les années 1975 et 1980 les industriels étaient très marqués par le développement de l'électronique et en particulier des bascules *RS* (*Reset/Set*) et par leur usage probable dans le cadre des automatismes séquentiels (en fait les Automates Programmables Industriels se sont développés plus vite que prévu). Ces bascules ont deux états (0 et 1) et permettent de mémoriser que l'on est dans une certaine étape d'une séquence ou non (une bascule est associée à chaque étape).

2 Du réseau de Petri au Grafcet

Comme nous l'avons déjà mentionné, le Grafcet peut être considéré comme un réseau de Petri *interprété*. Pour retrouver éventuellement le réseau de Petri sous-jacent à un Grafcet, il faut d'abord *enlever tout ce qui concerne la spécification des réceptivités, celle des actions ainsi que tout ce qui concerne le temps*.

Il reste donc un graphe comprenant des étapes et des transitions que l'on peut voir comme les places et les transitions d'un réseau de Petri sous-jacent, mais il s'agit d'un réseau de Petri particulier. En effet, une étape peut être active ou inactive ce qui correspond au fait qu'une place d'un réseau de Petri peut être vide ou contenir un jeton. Mais, dans le cas général, une place d'un réseau de Petri peut contenir plus d'un jeton. La classe particulière de réseau de Petri correspondant au Grafcet est celle des réseaux de Petri *saufs* (chaque place ne peut pas recevoir plus d'un jeton).

De ce point de vue on peut considérer que *le Grafcet est une norme pour l'interprétation des réseaux de Petri sauf dans le contexte des automatismes séquentiels*.

3 Différences entre le Grafcet et les réseaux de Petri

Les industriels voulant pouvoir implémenter directement un Grafcet en associant une bascule RS à chaque étape, un certain nombre de différences ont été introduites.

3.1 Activation d'une étape active

Le principe des bascules RS est d'avoir une entrée pour le signal Set et une entrée distincte pour le signal $Reset$. Ces entrées ne prennent en compte les signaux qu'au moment des impulsions de l'horloge. Si l'on envoie une impulsion sur l'entrée Set alors que la bascule est déjà à 1, elle reste dans cet état.

De la même façon, si une étape d'un Grafcet est activée par le franchissement de l'une de ses transitions amont elle devient active si elle était inactive et reste active sinon.

Par contre dans un réseau de Petri, si une transition d'entrée d'une place contenant un jeton est franchie, le contenu de la place est augmenté de 1 (ou de k si le poids de l'arc est k). Une place d'un réseau de Petri est un *compteur*, pas une *bascule*.

Le fait qu'un réseau de Petri soit sauf est une *propriété* qu'un réseau peut posséder ou non. Si le réseau de Petri n'est pas sauf (c'est-à-dire si certaines places peuvent contenir plus d'un jeton pour certains marquages accessibles) alors que l'on avait décidé de représenter un automatisme séquentiel, cela veut dire que *l'on a fait une erreur*. Il y a nécessairement un certain nombre de contradictions au sein de la représentation. Vérifier que le réseau de Petri est sauf est un élément de la validation des spécifications.

Le Grafcet corrige en quelque sorte automatiquement l'incohérence puisque si l'on active une étape active elle reste simplement active. Mais cette correction peut très bien ne pas correspondre au fonctionnement désiré. Lorsque l'on souhaite vérifier avec soin (mais ce n'est pas obligatoire) la cohérence d'un Grafcet, on vérifie qu'aucune étape active n'est activée (ce comportement est jugé dangereux) ce qui permet de retrouver la démarche effectuée dans le cas des réseaux de Petri.

3.2 Synchrone et asynchrone

Un modèle d'un système (système à événements discrets) est dit *synchrone* lorsqu'il comporte une *horloge de base* et que les événements ne peuvent se produire (ou bien qu'ils ne peuvent être *pris en compte*) qu'aux instants définis par les impulsions de cette horloge de base. Le temps est donc échantillonné par ces impulsions. Il peut arriver que plusieurs événements se produisent (ou soient pris en compte) lors de la même impulsion. Ils sont alors vus comme simultanés.

Considérons la figure 1. Les seules dates possibles pour un système synchrone sont t_{n-1} , t_n , t_{n+1} , t_{n+2} etc. L'événement Ev_i est alors associé au temps t_n , début du traitement $trait_i$ pendant lequel il est pris en compte. Les deux événements Ev_j et Ev_k sont vus comme simultanés et se produisant à la date t_{n+1} . Ils seront traités par $trait_{jk}$. Aucun événement n'est associé à t_{n+2} .

Les bascules RS sont synchrones puisque pour éviter les aléas qui pourraient être introduits par les temps de propagation des signaux électriques, les entrées $Reset$ et Set ne sont possibles que lors des impulsions de l'horloge de base (ces impulsions sont suffisamment espacées de façon à garantir que les bascules sont dans un état stable quand l'impulsion suivante arrive).

Le Grafcet est un modèle *synchrone*. On suppose que le temps est échantillonné ce qui est

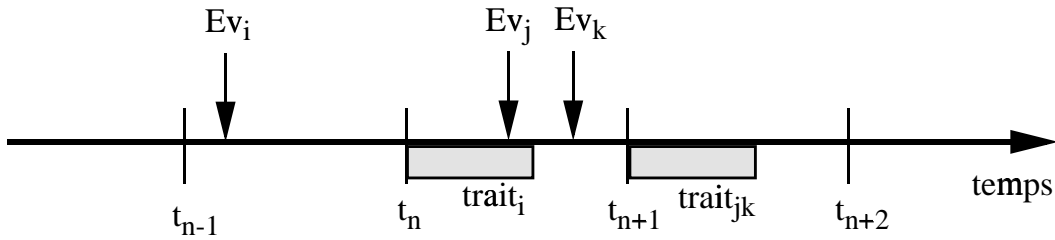


Figure 1: Système synchrone

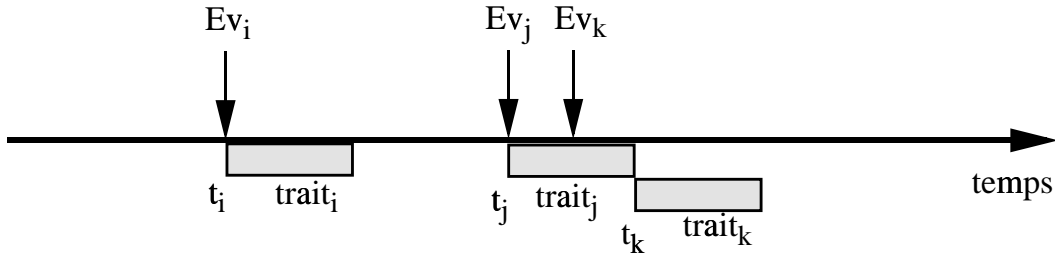


Figure 2: Système asynchrone

réaliste car les valeurs des capteurs ne sont lues par l'automate programmable que lors de certaines impulsions de l'horloge de base (ou de celles du réseau local de type *bus de terrain* lorsque les entrées sont déportées). On suppose en général que les valeurs des capteurs sont mises à jour en début de *cycle*. Plus d'un capteur peut changer de valeur à un instant donné et donc plus d'une transition peut être validée. *L'ensemble des transitions validées sont simultanément franchies.*

Un modèle d'un système (système à événements discrets) est *asynchrone* lorsqu'il ne fait aucune référence à une horloge de base. Le temps est considéré comme une variable continue (*dense*, c'est-à-dire représentée par un nombre réel ce qui implique qu'entre deux instants il est toujours possible de définir une infinité d'instants intermédiaires). Les événements peuvent se produire à n'importe quel instant et comme ils sont de durée nulle, il n'est pas possible que deux événements se produisent au même instant (même s'ils sont infiniment proches).

Si l'on considère la figure 2, les événements sont pris en compte comme ils arrivent. En général (en considérant que tous les traitements ont même degré de priorité), un nouvel événement ne peut être traité que si le traitement précédent est terminé. Les trois événements sont donc pris en compte aux instants t_i , t_j et t_k . Même si les événements Ev_j et Ev_k sont très proches, ils ne sont pas considérés comme simultanés et sont donc pris en compte l'un après l'autre.

Le réseau de Petri est un modèle *asynchrone*. Il n'est fait référence à aucune horloge, à aucune notion d'impulsion. Lors de la définition du comportement dynamique, on suppose toujours qu'une seule transition est franchie à chaque instant. Quand on construit le graphe des marquages accessibles [1], les arcs sont toujours étiquetés par le nom d'une seule transition. On ne représente pas des changements d'état qui correspondraient au franchissement simultané de plusieurs transitions.

3.3 Notion de conflit

Les différences de comportement du Grafset et des réseaux de Petri en cas de conflit découlent directement de la nature synchrone de l'un et asynchrone de l'autre.

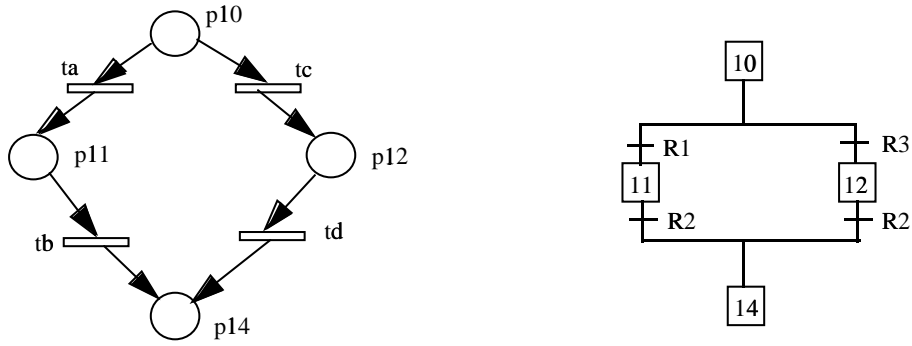


Figure 3: Exemple de conflits

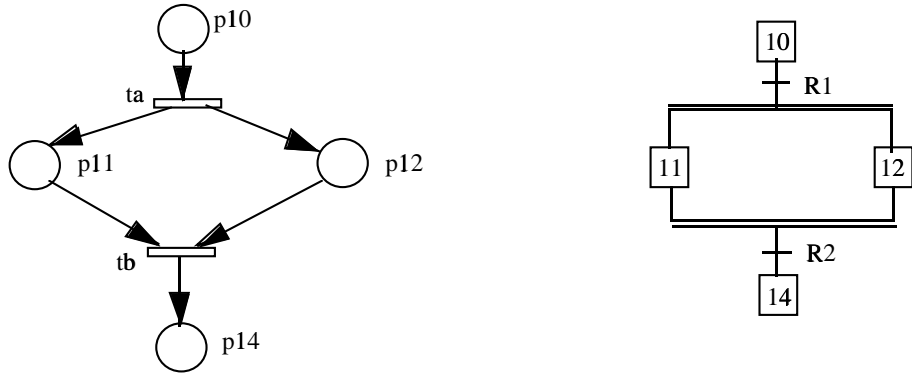


Figure 4: Exemple de parallélisme

Dans un réseau de Petri il y a conflit lorsqu'un marquage sensibilise plus d'une transition, mais que le contenu en jetons des places est insuffisant pour que toutes ces transitions soient effectivement franchies.

C'est le cas des transitions t_a et t_c dans la figure 3 si la place p_{10} contient un unique jeton. Comme le réseau de Petri est un modèle asynchrone, les événements associés aux franchissements de t_a et de t_c ne peuvent se produire au même instant. Une transition sera d'abord franchie et l'autre deviendra alors non franchissable. En conséquence, seuls deux marquages peuvent suivre un marquage M tel que :

$$M(p_{10}) = 1 \text{ et } M(p_{11}) = 0 \text{ et } M(p_{12}) = 0$$

Ce sont :

- soit M_1 avec $M_1(p_{10}) = 0$ et $M_1(p_{11}) = 1$ et $M_1(p_{12}) = 0$,
- soit M_2 avec $M_2(p_{10}) = 0$ et $M_2(p_{11}) = 0$ et $M_2(p_{12}) = 1$.

Il est impossible d'atteindre un marquage M_3 avec $M_3(p_{11}) = 1$ et $M_3(p_{12}) = 1$.

Au contraire dans un Grafcet, comme toutes les transitions validées sont franchies simultanément, si, dans la figure 3 l'étape 10 est active et si les deux réceptivités R_1 et R_3 sont toutes les deux vraies au temps (discret) considéré, alors la situation suivante sera telle que les deux étapes 11 et 12 sont toutes les deux actives (10 devenant inactive). Bien entendu le Grafcet a le même comportement que le réseau de Petri si seule l'une des deux réceptivités R_1 ou R_3 est vraie.

Si l'on souhaite, dans le cadre d'un réseau de Petri, avoir les deux places p_{11} et p_{12} simultanément marquées et passer de M à M_3 ($M_3(p_{11}) = 1$ et $M_3(p_{12}) = 1$), alors il faut choisir le réseau de Petri de la figure 4, mais les marquages M_1 et M_2 ne seront plus des marquages accessibles à partir de M . Le Grafcet de cette figure possède le même comportement.

Si, chaque fois que deux transitions partagent la même étape d'entrée on vérifie (pour des raisons de sécurité) qu'il est absolument impossible que les réceptivités associées (R_1 et R_3 sur la figure 3) soient simultanément vraies, alors les comportements des réseaux de Petri et des Grafcets seront les mêmes dans tous les cas.

4 Conclusion

Résumons en quelques mots. Par le choix de l'approche *synchrone* pour le Grafcet, ce modèle se différencie fortement du réseau de Petri et de son comportement *asynchrone*. Toutefois, pour rendre la vérification et la validation de l'automatisme séquentiel plus facile, il est fréquent que l'on restreigne les possibilités du Grafcet (ne pas activer une étape déjà active et vérifier que les réceptivités ne sont pas vraies simultanément) et qu'alors sa dynamique devienne identique à celle du réseau de Petri sauf correspondant. Il est, dans ce cas, possible de faire une première spécification abstraite avec un réseau de Petri sauf, puis, après une première validation fonctionnelle, de passer au Grafcet pour expliciter les interactions avec les capteurs et les actionneurs. Cela permet de une mise en œuvre directe sur un automate programmable industriel au fonctionnement synchrone.

En ce qui concerne les domaines d'application, le Grafcet est bien adapté à la programmation d'un automate directement en interaction avec les capteurs et les actionneurs (commande locale). Le Grafcet est plus proche de la mise en œuvre car il est synchrone et on spécifie les réceptivités et les actions en même temps que l'on construit le graphe.

Le réseau de Petri est plus adapté pour spécifier un système distribué (on ne sait pas encore comment les fonctions seront réparties sur un ensemble d'automates et donc on ne peut pas encore exploiter la nature synchrone du fonctionnement de chaque automate).

Il est également mieux adapté à la spécification de fonctions de supervision car on a souvent besoin de représenter des stocks (un compteur est alors plus adapté qu'une bascule) et car ces fonctions interagissent avec les automates de la commande locale par des échanges asynchrones de messages. La supervision n'est pas directement en relation avec les capteurs et les actionneurs. Un superviseur reçoit un flot de messages, de la part des automates, qu'il traite les uns après les autres. Il envoie un flot de messages vers les automates pour changer certains paramètres, par exemple. Les contraintes temporelles étant plus faibles que pour la commande locale, on peut traiter complètement un message avant de prendre en compte le message suivant.

References

- [1] R. Valette. Les réseaux de Petri. Cours photocopié, septembre 2000.
<http://www.laas.fr/~robert/cour96.pdf>