Verification quantitative des modèles paramétrés

Aina Toky RASOAMANANA

January 23, 2017

Encadrants : Nathalie BERTRAND et Nicolas MARKEY

IRISA équipe SUMO

1 Introduction

Les modèles paramétrés permettent de représenter naturellement des systèmes formés d'un grand nombre, souvent inconnu, de composants identiques. Pour valider ces systèmes, une option naïve consiste à fixer une limite sur le nombre de processus et à appliquer des techniques classiques de vérification. Une autre option est d'utiliser des techniques de vérification paramétrée. Ces techniques visent à valider le type de systèmes definit ci-dessus, indépendamment de l'instance précise du modèle, c'est-à-dire indépendamment du nombre de composants. Non seulement, cette dernière approche est plus génerale, mais elle peut également se révéler plus efficace. Les protocoles distribués sont un exemple motivant la vérification paramétrée ; en effet, leur bon fonctionnement ne doit pas dépendre du nombre de participants. On peut citer d'autres cas d'application : les programmes multi-thread, ainsi que certains systèmes biologiques ou chimiques.

Des algorithmes ont été proposés récemment pour vérifier certaines propriétés qualitatives (par exemple la sûreté ou la vivacité) pour des modèles paramétrés tels que les protocoles de population [2] et les réseaux d'automates avec variables partagées [1].

La section 2 présente l'etat de l'art. Elle décrit le modèle considéré lors du stage ainsi que la vérification à taille fixée et la vérification paramétrée. La section 3 présente les objectifs du stage.

2 Etat de l'art

2.1 Définition du modèle

Definition 1. Un protocole avec registre est un quadruplet $P = (Q, D, q_0, T)$, avec Q c'est l'ensemble fini des emplacements de controle, D un alphabet fini de données, $q_0 \in Q$ l'emplacement initial et $T \subseteq Q \times \{R, W\} \times D \times Q$ l'ensemble des transitions du protocole. R signifie lire le contenu du registre et W signifie ecrire sur le registre.

Example 2. Prenons $D = \{0, 1, 2\}$ et $Q = \{q_0, q_1, q_2, q_f\}$

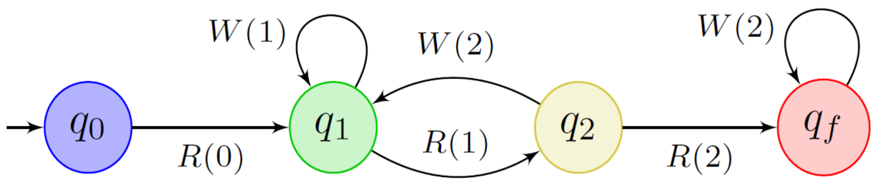

Figure 2.1: Exemple de protocole avec registre pour $D = \{0, 1, 2\}$

Dans cette figure, lorsque le registre contient 0, ce protocole peut passer de q_0 à q_1 en lisant 0 dans le registre. De q_1, il peut écrire 1 dans le registre puis lire 1 et passer dans q_2. A partir de q_2, le processus peut rester dans q_2 ou écrire 2 dans le registre et passer dans q_1.

On note M l'ensemble des multi-ensembles sur Q tel que $M = \{\mu : Q \longrightarrow \mathbb{N}\}$ et M_N l'ensemble des multi-ensembles sur Q de taille N : $M_N = \{\mu \in M \mid \sum_{s \in Q} \mu(s) = N\}$. Soit μ et μ' deux multi-ensembles. On dit que μ est inclus dans μ' (noté $\mu \sqsubseteq \mu'$) si pour tout $s \in Q$, $\mu(s) \leq \mu'(s)$. De plus, $\mu + \mu'$ est un multi-ensemble tel que pour tout $s \in Q$, $(\mu + \mu')(s) = \mu(s) + \mu'(s)$. Supposons que $\mu \sqsubseteq \mu'$, $\mu' - \mu$ est un multi-ensemble tel que $(\mu' - \mu)(s) = \mu'(s) - \mu(s)$.

Definition 3. **Une configuration** dans un protocole avec registre est un élement de $\Gamma = M \times D$. On note Γ_N, l'ensemble des configurations à N copies du protocole, $\Gamma_N = M_N \times D$.

Une exécution finie de N protocoles est une suite de configuration $\gamma_0, \gamma_1, ..., \gamma_k$ tel qu'il existe une transition entre γ_i et γ_{i+1} pour tout $i \in \{0, ..., k-1\}$. Notons par $Exec_N$ l'ensemble des exécutions finies de N protocoles.

Le graphe d'éxecution de N copies du protocole est un quadriplet $G_e = (\Gamma_N, D, \gamma_0, T_N)$ associé au protocole avec registre P où:

- $\gamma_0 = \left(q_0^N, d_0\right)$: configuration initiale

- $T_N \subseteq \Gamma_N \times \{R, W\} \times D \times \Gamma_N$ et $t_N = ((\mu, d), A, d'', (\mu', d')) \in T_N$ si $\exists t = (q, A, d'', q') \in T$ tel que

- $\mu(q) > 0$
- $\mu - q + q' = \mu'$
- $d'' = d' = d$ si $A = R$ et $d'' = d'$ si $A = W$

Example 4. Prenons $N = 2$.

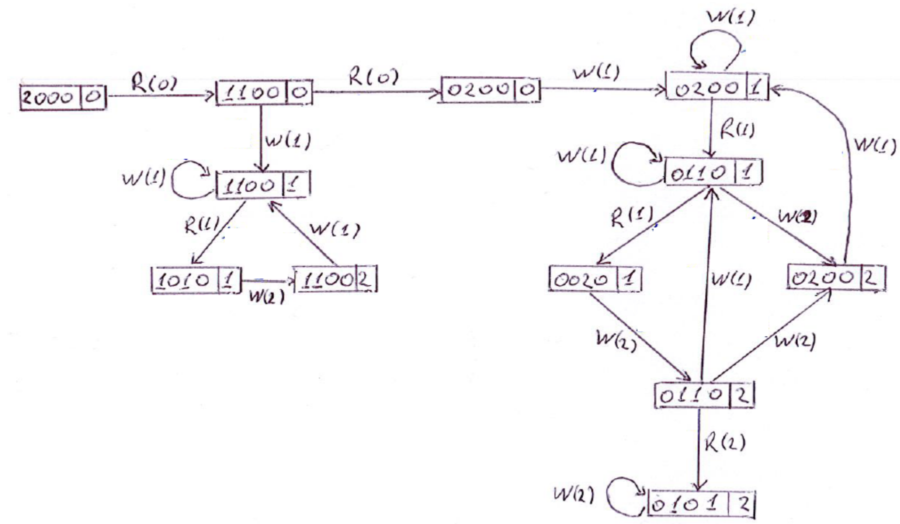

Figure 2.2: Graphe d'exécution pour $N = 2$ copies du protocole défini dans la figure 2.1

Initialement, les deux protocoles sont dans q_0 et la valeur du registre vaut 0. De la configuration initiale $(2000 \mid 0)$, l'un des deux protocoles peut lire 0 dans le registre et passe dans l'état q_1, d'où la configuration $(1100 \mid 0)$. De cette dernière configuration, si le protocole, qui est dans l'état q_0, lit 0 dans le registre, alors on passe de $(1100 \mid 0)$ à $(0200 \mid 0)$. Par contre, si l'autre protocole écrit 1 dans le registre, alors on passe de $(1100 \mid 0)$ à $(1100 \mid 1)$.

Il est important d'aborder la notion de scheduler pour determiner quel système va faire une telle transition.

Definition 5. Scheduler deterministe, scheduler aléatoire

Un scheduler (aléatoire) est une fonction $f_N : Exec_N \longrightarrow Distr\left(\{1, ..., N\}, T\right)$ tel que si γ est la dernière configuration d'une exécution ρ, si de plus, $(n, t) \in f_N(\rho)$, alors le protocole n peut faire la transition t depuis γ.

En particulier, un scheduler deterministe est une fonction $f_N : Exec_N \longrightarrow \{1, ..., N\} \times T$.

Notation : $Exec_N(f_N)$ l'ensemble des exécutions compatible avec un scheduler f_N.

Remark 6. Si f_N est deterministe, alors le cardinal de $Exec_N(f_N)$ vaut 1.

Example 7. Reprenons la figure 2.2.

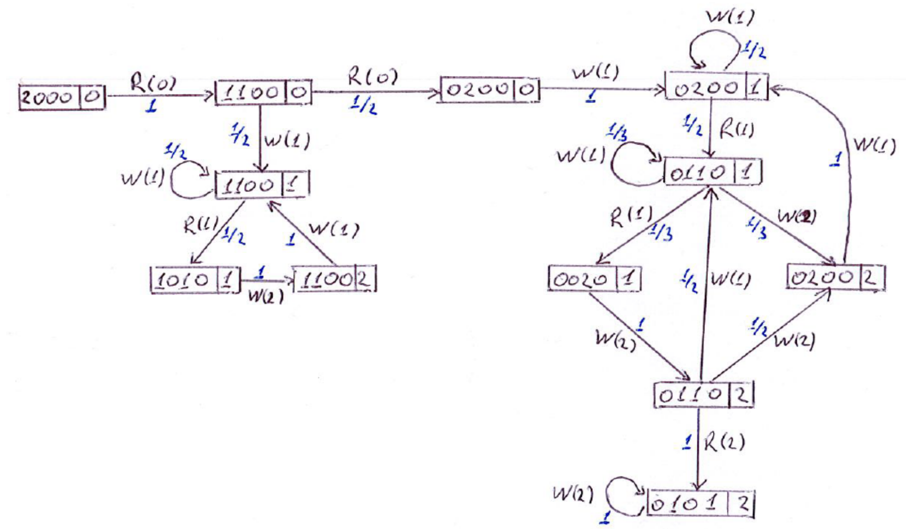

Figure 2.3: Graphe d'exécution à 2 copies du protocole défini dans la figure 2.1 associé à un scheduler uniforme aléatoire.

Cette figure illustre les probabilités de choisir chacune des transitions à une configuration bien précise. Par exemple, à partir de la configuration $(0110 \mid 1)$, on peut accéder aux configurations $(0110 \mid 1)$, $(0020 \mid 1)$, $(0200 \mid 2)$ avec une même probabilité égale à $\frac{1}{3}$ en supposant un scheduler aléatoire qui choisit uniformement une transition possible.

2.2 Verification à taille fixée

Soit $N \in \mathbb{N}$. On note $\Diamond q_f$ l'ensemble de toutes les exécutions allant de γ_0 à une configuration dans $F_N = \{(\mu, d) \in \Gamma_N \mid \mu(q_f) > 0\}$.

Definition 8. Problèmes de décision à N fixé

Nous nous interessons aux problèmes suivants :

(P_1) Existe-t-il un scheduler f_N deterministe tel qu'on atteint une configuration finale dans F_N ? Autrement dit, existe-t-il un scheduler f_N deterministe tel que $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$?

(P_2) Est-ce que, pour tout scheduler f_N deterministe, on atteint une configuration finale dans F_N? Ou encore, pour tout scheduler f_N deterministe, a-t-on $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$?

(P_3) Soit f_N le scheduler uniforme aléatoire et A_{N,f_N} la chaine de Markov associée au scheduler f_N. Notons par $P_{A_{N,f_N}}(X)$ la probabilité de X engendrée par A_{N,f_N}. A-t-on $P_{A_{N,f_N}}(\Diamond q_f \cap Exec_N(f_N)) = 1$?

On peut caractériser sous forme de problème de théorie des graphes les problèmes définis précedemment.

Proposition 9. *(P_1) est équivalent à " existe-t-il un chemin dans G_e de la configuration initiale γ_0 à une configuration finale dans F_N ? "*

(P_2) est équivalent à " tout chemin dans G_e mène-t-il à une configuration finale F_N ? "

(P_3) est équivalent à " Est-ce que toutes les composantes connexes terminales contiennent une configuration finale F_N ? "

Example 10. Considérons la figure 2.3.

Remarquons que le chemin $(2000 \mid 0) \longrightarrow (1100 \mid 0) \longrightarrow (0200 \mid 0) \longrightarrow (0200 \mid 1) \longrightarrow (0110 \mid 1) \longrightarrow (0200 \mid 2) \longrightarrow (0110 \mid 2) \longrightarrow (0101 \mid 2)$ est un chemin qui part de $\gamma_0 = (2000 \mid 0)$ vers une configuration finale $\gamma_f = (0101 \mid 2)$. D'où l'existence d'un scheduler deterministe, noté f_2, tel que $\Diamond q_f \cap Exec_2(f_2) \neq \emptyset$ c'est à dire que la réponse au problème (P_1) est " vrai ".

Remarquons de plus que, tous les chemins commencant par $(2000 \mid 0) \longrightarrow (1100 \mid 0) \longrightarrow (1100 \mid 1) \longrightarrow ...$, ne pourront jamais atteindre une configuration finale. D'où l'existence d'un scheduler deterministe, noté f, tel que $\Diamond q_f \cap Exec_2(f) = \emptyset$, c'est à dire que la réponse au problème (P_2) est " faux ".

En outre, pour le scheduler aléatoire f' choisi à la figure 2.3,
$P_{A_{N,f_N}}(\Diamond q_f \cap Exec_2(f')) \leq \frac{1}{2}$. Donc, la réponse au problème (P_3) est " faux ".

En analysant de près le problème (P_1), on remarque une propriété de monotonie.

Proposition 11. *Si (P_1) est vraie pour $N \in \mathbb{N}$, alors (P_1) est vraie pour tout $M \geqslant N$.*

Idée pour la preuve : Supposer que (P_1) est vraie pour $N \in \mathbb{N}$. Soit $M \in \mathbb{N}$ tel que $M \geqslant N$. Il se peut que $M - N$ protocoles restent dans l'état q_0 et N protocoles seulement bougent. Dans ce cas, on retrouve toutes les propriétés associées à celui du graphe d'exécution à N copies du protocoles.

2.3 Vérification paramétrée

Soit $P = (Q, D, q_0, T)$ un protocole avec registre et $d_0 \in D$.

Definition 12. Le graphe symbolique associé à P et d_0 est un triplet $G = (V, v_0, E)$ où $V = 2^Q \times D$ est l'ensemble des sommets du graphe G, $v_0 = (\{q_0\}, d_0)$ et $E \subseteq V \times V$ tel que $t = ((S, d), (S', d')) \in E$ avec $S, S' \subseteq Q$ si il existe une transition $(q, A, d'', q') \in T$ tel que $d = d' = d''$ si $A = R$ et $d' = d''$ si $A = W$ et

- soit $S' = S$ si $q' = q$
- soit $S' = \{\{S \setminus \{q\}\} \cup \{q'\}, S \cup \{q'\}\}$ si $q \neq q'$.

La taille du graphe symbolique est de $2^{|Q|} \times \mid D \mid$.

Example 13. Prenons le protocole P de l'exemple 2 de la figure 2.1

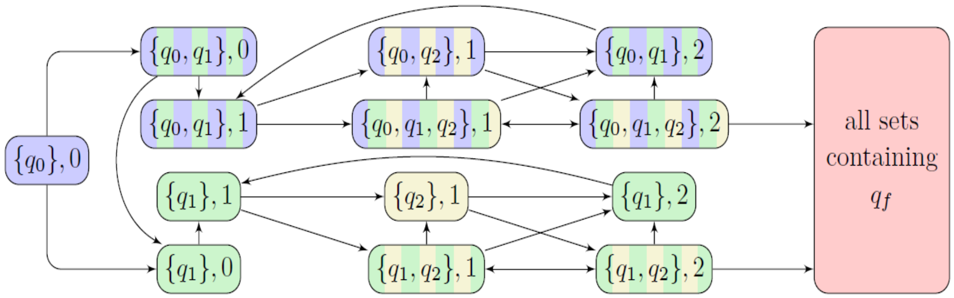

Figure 2.4: Graphe symbolique associé au protocole du figure 2.1

De la même manière que celui du graphe d'exécution à N copies du protocole, de $(\{q_0, q_2\}, 1)$, soit tous les protocoles à l'état q_2 écrivent 2 dans le registre, si c'est le cas, on passe dans $(\{q_0, q_1\}, 2)$ et soit certain protocoles à l'état q_2 écrivent 2 dans le registre et d'autre n'y écrivent pas, dans ce cas, on passe dans $(\{q_0, q_1, q_2\}, 2)$.

Definition 14. Problème de decision paramétrée (N non fixé)

(P_1') Existent-ils $N \in \mathbb{N}$ et un scheduler f_N deterministe tel que $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$?

(P_2') Pour tout $N \in \mathbb{N}$ et pour tout scheduler deterministe f_N, a-t-on $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$?

Pour résoudre (P_1') et (P_2'), on montre les caractérisations ci-dessous.

Proposition 15. *(P_1') est vrai si et seulement si il existe un chemin de $(\{q_0\}, 0)$ à un sommet (E, d), où $q_f \in E$, dans le graphe symbolique G.*

(P_2') est vrai si et seulement si tous les chemins de $(\{q_0\}, 0)$ atteignent un sommet (E, d) tel que $q_f \in E$ dans le graphe symbolique G.

Le graphe symbolique n'est pas suffisant pour résoudre la variante paramétrée du problème (P_3). Cette variante paramétrée de (P_3) a été étudiée dans [1].

3 Objectifs du stage

Tout au long de notre stage, on voudrait faire une analyse quantitative des problèmes définis dans la section 2. Dans cette section, c'est à dire la section 2, on a definit deux catégories de problèmes différents : les problèmes à N fixé et les problèmes paramétrés.

3.1 Vérification à taille N fixé

Dans cette section, on s'interesse aux trois questions suivantes :

- Si on suppose qu'il existe un scheduler deterministe tel qu'à partir de la configuration initiale, on atteint une configuration finale dans F_N ou bien il existe un scheduler deterministe f_N tel que $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$, alors on définit la question (Q_1) : " Quel est le scheduler qui minimise le nombre d'étapes pour atteindre q_f ? ".

- Si on suppose que pour tout scheduler deterministe, on atteint une configuration finale dans F_N à partir d'une configuration initiale ou encore $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$ pour tout scheduler f_N deterministe, alors on définit la question (Q_2): " Quel est le scheduler qui maximise le nombre d'étapes pour atteindre q_f ? ".

- Si on suppose que la réponse au problème (P_3) est positive, on définit la question Q_3: " Quel est le nombre d'étapes moyen pour atteindre q_f ?".

En analysant de près les questions (Q_1) et (Q_2) qu'on vient de définir, résoudre la question (Q_1) (respectivement (Q_2)) revient à trouver un plus court chemin (respectivement un plus long chemin) de la configuration initiale γ_0 à une configuration finale dans F_N, dans un graphe d'exécution G_e à N copies du protocole. On connait déjà des algorithmes naïfs pour résoudre un tel problème. Par contre, la taille du graphe G_e est exponentielle $\left(\mid Q \mid^N\right)$. On voudrait donc trouver des algorithmes plus efficaces, sans construire le graphe G_e, pour trouver un plus court chemin et un plus long chemin de la configuration initiale γ_0 à une configuration finale dans F_N. De même pour la question (Q_3), il existe aussi des algorithmes pour calculer le temps moyen sur le graphe G_e, mais encore une fois cela nécessite de calculer le graphe G_e. Donc, on envisagerait une approche analytique pour pouvoir calculer le temps moyen pour atteindre les configurations finales dans F_N.

3.2 Vérification paramétrée

On s'interesse aux problèmes suivants :

- Si on suppose qu'ils existent $N \in \mathbb{N}$ et un scheduler f_N deterministe tel que $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$, on définit (Q'_1): " Calculer le temps le plus court f_1 en fonction de N où f_1 est une fonction qui associe le nombre minimal d'étapes pour atteindre q_f ". Tout de suite, on peut remarquer que f_1 est une fonction partielle puisqu'on ne suppose pas qu'il existe un scheduler pour chaque N satisfaisant $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$. Pour certaines valeurs de N, il n'existera pas de scheduler correct ou encore $\exists N \in \mathbb{N}$ tel que pour tout scheduler f deterministe, $\Diamond q_f \cap Exec_N(f) = \emptyset$. On voudra donc aussi calculer le domaine de définition de f_1, c'est-à-dire l'ensemble des N pour lesquels f_1 est définie.

- Si on suppose que pour tout $N \in \mathbb{N}$ et pour tout scheduler f_N deterministe, on a $\Diamond q_f \cap Exec_N(f_N) \neq \emptyset$, on définit (Q'_2): " Calculer une fonction f_2, dépendant de N, qui associe le nombre maximal d'étapes pour atteindre q_f ".

- Si on suppose que la réponse au problème (P_3) est positive, on définit (Q'_3): " Calculer une fonction f_3 qui associe le nombre moyen d'étapes pour atteindre q_f ".

Notre objectif pendant le stage sera d'étudier les problèmes de vérifications quantitatives (Q_1), (Q_2), (Q_3), (Q'_1), (Q'_2) et (Q'_3) qu'on vient de définir.

References

[1] P. Bouyer, N. Markey, M. Randour, A. Sangnier, and D. Stan. Reachability in networks of regis- ter protocols under stochastic schedulers. In I. Chatzigiannakis, M. Mitzenmacher, Y. Rabani, and D. Sangiorgi, editors, Proceedings of the 43rd International Colloquium on Automata, Languages and Programming (ICALP'16) – Part II, volume 55 of Leibniz International Proceedings in Informatics, pages 106 :1–106 :14. Leibniz-Zentrum für Informatik, July 2016.

[2] J. Esparza, P. Ganty, J. Leroux, and R. Majumdar. Verification of population protocols. In L. Aceto and D. de Frutos-Escrig, editors, Proceedings of the 26th International Conference on Concurrency Theory (CONCUR'15), volume 42 of Leibniz International Proceedings in Informatics, pages 470–482. Leibniz-Zentrum für Informatik, Sept. 2015.

[3] M. Ummels and Ch. Baier. Computing quantiles in markov reward models. In F. Pfenning, editor, Proceedings of the 16th International Conference on Foundations of Software Science and Computation Structure (FoSSaCS'13), volume 7794 of Lecture Notes in Computer Science, pages 353–368. Springer-Verlag, Mar. 2013.