Une technique de validation de protocoles basée sur une exploration avec retour arrière des automates complexes

Farah Zoubeyr[1], Abdelkamel Tari[1], and Zahir Tari[2]

[1] Département d'Informatique, Université Abderrahmane Mira de Béjaïa, Route de Targa Ouzemmour Béjaïa 06000, Algérie.
zobifara@gmail.com, tarikamel@yahoo.fr

[2] Royal Melbourne Institute of Technology (RMIT), Department of Computer Science, Melbourne, Australia.
zahirtari@rmit.edu.au

Résumé L'analyse d'accessibilité est généralement l'approche utilisée pour la validation des protocoles. Le problème posé par l'application de cette technique est celui de l'explosion combinatoire. Dans cet article, nous proposons une nouvelle technique de validation de protocoles appelée Reverse Leaping Reachability Analysis (RLRA). Au lieu de parcourir tout l'espace des états de protocole, notre approche explore seulement un ensemble réduit de cet espace afin de détecter les erreurs de type Interblocage. Les résultats d'expérimentation ont montré que notre technique permet de réduire la taille du graphe d'accessibilité de protocole est donc réduire le problème de l'explosion combinatoire.
Mots-clé: validation de protocoles, automate à états complexes, interblocage

1 Introduction

Dans le domaine des Services Web, les protocoles de communication peuvent être modélisés sous forme de communication d'automates à états finis. Dans ce modèle, le comportement de chaque entité communicante est spécifié par un Automate à Etats Finis (AEF). Ces entités échangent des messages via des canaux de communication de capacité limitée. La validation de protocoles, consiste à vérifier l'absence des erreurs logiques de spécification telles que les erreurs d'interblocage et les réceptions non spécifiées. L'analyse d'accessibilité standard est généralement l'approche la plus utilisée dans le domaine de validation de protocoles. Le principe de cette technique est l'énumération de toutes les interactions possibles entre les entités communicantes. Cette énumération génère des états globaux de protocole où chaque état global donne la situation du protocole après avoir exécuté une opération. A la fin de cette opération, on obtient un graphe d'accessibilité de protocole. Les erreurs de protocole, sont alors détectées par la vérification de chaque état global composant ce graphe. L'analyse d'accessibilité standard est une technique intuitive, simple et permet la détection de toutes les erreurs que peut contenir une spécification de protocole. Cependant, elle soufre

du problème de l'explosion combinatoire. En général, le nombre total des états globaux d'un protocole est exponentiel en terme de nombre d'entité composant le protocole ainsi que le nombre d'état local de chaque entité [4]. Bien que plusieurs versions (améliorations) ont étés proposées [1,2,3] pour allégir l'analyse d'accessibilité standard, le problème de l'explosion combinatoire était toujours présent, ce qui a limité leur application à des protocoles simples.

L'analyse d'accessibilité réversible [5] est une alternative intéressante de l'analyse d'accessibilité standard. A partir des propriétés qui caractérisent les erreurs de type interblocage, on dégage un ensemble d'états globaux qui vérifient ces propriétés. Ces états sont appelés états suspects. Par la suite, un état suspect est confirmé comme une erreur d'interblocage si l'état global initial du protocole est atteignable par une opération de retour arrière à partir de cet état suspect, sinon l'état suspect est retiré de l'ensemble des erreurs de protocole. Avec ce procédé, cette technique permet de parcourir seulement un sous-ensemble de l'espace des états du protocole. Bien que cette technique permet de réduire l'espace des états à explorer durant l'analyse, le problème de l'explosion combinatoire peut se présenter pour une validation de protocole complexe. Donc, pour rendre cette approche de validtion efficace nous avons besoin d'optimiser la construction des chemins de retour arrière.

Dans cet article, nous proposons une technique de validation de protocoles modélisés sous forme de Communication d'Automate d'Etats Finis Complexes (CAEFC), appelée RLRA (Reverse Leaping Reachability Analysis). Notre technique, permet la détection de toutes les erreurs de type interblocage dans une spécification de protocole. En se basant sur une approche de validation avec retour arrière, on commence par identifier de possibles états d'interblocage ensuite, valider parmi ces états suspects, ceux qui présentent réellement une erreur à travers la recherche de chemins de retour vers la racine du graphe de protocole. Pour réduire le problème de l'explosion combinatoire, nous introduisons les graphes de sauts (leap graphe) que nous exploitons dans la construction des chemins de retour arrière. Ces graphes de sauts, permettent d'éviter le parcourt des états globaux qui ne sont pas utiles dans l'analyse, ce qui apporte une réduction importante sur la taille du graphe à parcourir. Pour chaque erreur confirmée, nous donnons aussi sa nature (simple, hybride ou complexe) selon la nature des états qui présentent l'erreur.

Le reste de cet article est organisé comme suite : La prochaine section introduit le modèle de CAEFC. La section trois, explique notre approche de validation ainsi que les prouves formelles de son applicabilité. La quatrième section, donne les résultats d'expérimentation. La dernière section est une conclusion de ce travail.

2 Modèle de Communication d'Automates à Etats Finis Complexes (CAEFC)[9]

Un modèle de CAEFC introduit la notion d'état complexe, qui est un AEF dont certains de ses états sont eux-mêmes des AEF (internes). Les états qui

rentrent dans la composition d'un état complexe sont appelés états internes. Le développement des différents AEF internes donne un AEF aplati, qui est un AEF ordinaire.
Soit $I = \{1, 2, ..., n\}$ un ensemble fini d'indices, avec cardinal(I)≥2. Dans un modèle de Communication d'automate d'états finis, un protocole Π est une paire (P, L), où $P = \{P_i / i \in I\}$ est l'ensemble des n entités communicantes du protocole et $L \subseteq I \times I$ est la relation d'incidence non réflexive qui identifie un ensemble non vide de canaux de communication $\{C_{ij}/(i,j) \in L\}$. Chaque canal C_{ij} est une file FIFO, bornée et sans erreurs reliant l'entité P_i à l'entité P_j. Chaque entité P_i est un automate d'états finis : (S_i,$S0_i$,Ef_i,M_i,$\triangle_i$) (simple ou complexe), composé d'un ensemble d'états S_i , d'un ensemble d'états finaux Ef_i ,d'un alphabet de messages M_i ,d'un ensemble de transitions $\triangle_i$ et d'un état initial $S0_i$. L'ensemble des états S_i comprend des états simples et des états complexes. Chaque état complexe doit être défini par un AEF. Le contenu du canal C_{ij}, noté c_{ij}, est une suite de messages de M_i, i.e. $c_{ij} \in M_i^*$. Chaque canal peut contenir un nombre maximum de K (entier positif) messages. Une entité P_i envoie des messages sur son canal de sortie C_{ij} $(i, j \in I \wedge i \neq j)$ et reçoit des messages sur ses canaux d'entrées C_{ji} $(i, j \in I \wedge i \neq j)$. La relation de transition $\triangle_i$ est définie par :$\triangle_i : S_i \times M_i \rightarrow S_i$. Une transition $t = (s_i, \mu, s_i^{'}) \in \Delta_i$, définie à l'état local s_i de l'entité P_i, est dite transition d'envoi de message lorsque $\mu = -x$, indiquant un envoi d'un message $x\,(x \in M_i)$ par l'entité Pi sur le canal C_{ij}. t est dite transition de réception de message lorsque $\mu = +y$, indiquant une réception d'un message $y\,(y \in M_i)$ par l'entité P_i depuis le canal C_{ji}. Les signes - et + indiquent respectivement un envoi et une réception de message. Dans le cas où $s_i^{'}$ est un état complexe (représente un AEF interne), le résultat de la transition va mettre l'automate interne sur sont état initial, si s_i et aussi un état complexe, la transition ne peut être exécutée que si l'automate lui correspondant atteint sont état final.

Un état global G du protocole Π est représenté par une paire $(< s_i^G >_{i \in I}, < c_{ij}^G >_{(i,j) \in L})$, où s_i^G est l'état local de l'entité P_i et cij^G est le contenu du canal C_{ij} à l'état global G. L'état global initial est notée $G0 = (< s_i^{G0} >_{i \in I}, < c_{ij}^{G0} >_{(i,j) \in L})$, tel que $s_i^{G0} = S0_i$ et $c_{ij}^{G0} = \varepsilon$, pour tout $i \in I$ et $(i, j) \in L$.

On definie la fonction $act()$ comme suite [8] :

1. Pour une transition $t \in \Delta_i$, $act(t) = i$ est l'indice de l'entité auquel appartient la transition t,

2. Pour un ensemble de transition T, $act(T) = \{i \in I / T \cap \Delta_i \neq \varepsilon\}$ est l'ensemble des indices des entités qui ont au moins une transition dans T ;

Nous définissons également la fonction $Front(c_{ij}^G)$ qui renvoie le message en tête du canal c_{ij} à l'état global G.

2.1 Types d'erreur dans une spécification par modèle CAEFC

En plus de la nature de l'erreur logique, l'utilisation des automates d'états complexes nécessite la définition du type de l'erreur détectée qui peut être de type simple, lorsque chaque entité communicante se trouve dans un de ses états simples, complexe lorsque toutes les entités se trouvent dans un de leurs états complexes et hybride lorsque une ou plusieurs entités du modèle se trouvent dans un état simple et le reste des entités se trouvent dans des états complexes.

2.2 Validation de protocoles

La validation d'un protocole revient à faire une analyse de son graphe d'accessibilité. Un graphe d'accessibilité donne les différentes exécutions possibles du protocole, les noeuds de ce graphe sont les états globaux du protocole et les arcs sont les différentes transitions (interactions) qui font passé le protocole d'un état global vers un autre. L'approche de validation exhaustive soufre du problème de l'explosion combinatoire qui est dû au grand nombre d'état global à générer pendant l'analyse. Ce problème peut être réduit en adoptant une approche de validation avec retour arrière. Cette approche est appelée analyse d'accessibilité réversible (RRA)[5]. Dans cette approche, on génère l'espace des états de protocole d'une manière réversible. A partir d'un ensemble d'états, dit indésirable, on essaye d'accéder l'état global initial par une opération de retour arrière. L'ensemble des états indésirables est composé des états globaux qui présentent des caractéristiques d'un état d'interblocage (chaque entité est dans un état d'attente de réception d'un message et l'état de leurs canaux d'entrée est vide).

3 Une technique de validation de protocoles pour les automates complexes

Dans cette section, nous proposons une technique de validation de protocoles qui est basée sur une approche avec retour arrière. Dans notre approche de retoure arrière, au lieu de travailler sur un graphe d'accessibilité standard (couvre tout l'espace des états), nous utilisons un graphe de sauts (LRA[3])[8]. Ce graphe est construit à partir d'un sous ensemble de l'espace des états de protocole. La particularité de ce graphe c'est qu'il couvre aussi tous les états qui peuvent présenter une erreur de type interblocage. Donc, l'utilisation d'un graphe de sauts réduit le nombre d'état à parcourir tout en gardant la capacité de détecter toutes les erreurs d'interblocage, ce qui permet de réduire la complexité du l'analyse de protocole.

3.1 Graphe de sauts [8]

Un graphe de sauts est un graphe d'accessibilité réduit. Cette réduction est obtenue du fait que le passage d'un état global vers un autre peut se faire via

[3] LRA :Leaping Reachability Analysis.

l'exécution de plusieurs transitions à la fois, ce qui permet d'éviter de générer des états intermédiaires résultants de l'exécution de chaque transition à part. les transitions qui peuvent s'exécuter en même temps sont appelées : transitions simultanément exécutables.Ces transitions simultanées forme un saut dans un graphe d'accessibilité.

Transitions simultanément exécutables. Pour calculer les transitions simultanément exécutables qui rentrent dans la construction d'un graphe de sauts, nous avons besoin d'abord de définir les notions de transitions exécutables et transitions potentiellement exécutables.

Transitions exécutables : Une transition $t = (s_i, \mu, s_i^{'})$ d'une entité communicante P_i est dite exécutable à l'état global G, ssi $s_i = s_i^G$ (i.e s_i est l'état local de l'entité P_i à l'état global G) et t vérifie une des deux conditions suivantes :

1. t est une transition d'envoi de message et le contenu du canal C_{ij} est inférieure à K, i.e. $|C_{ij}| < K$. ou,
2. t est une transition de réception de message tel que $\mu = +y$ avec $y \in M_i$ et $Front(c_{ji}^G) = y$. Autrement dit, t peut recevoir le message qui est en tête du canal c_{ji}.

L'ensemble des transitions d'envoi ou de réception de messages de l'entité P_i, qui sont exécutables à l'état globale G, sont notés respectivement $X_i^-(G)$ et $X_i^+(G)$. L'union de ces deux ensembles est noté $X_i(G) = X_i^-(G) \cup X_i^+(G)$. L'ensemble de toutes les transitions des différentes entité communicantes du protocole qui sont exécutables à l'état G, est noté : $X(G) = \cup_{i \in I} X_i(G)$.

Transitions potentiellement exécutables : Une transition de réception de message $t = (s_i, +y, s_i^{'})$ est dite potentiellement exécutable à l'état global G, ssi $s_i = s_i^G$ et $c_{ji}^G = \varepsilon$. Une transition d'envoi de message $t = (s_i, -x, s_i^{'})$, est dite potentiellement exécutable à l'état global G, ssi $s_i = s_i^G$ et $|C_{ij}| = K$; K est la capacité des canaux de communication du système. L'ensemble des transitions de réception de message et des transitions d'envoi de message de l'entité P_i qui sont potentiellement exécutables à l'état globale G, sont noté respectivement $P_i^+(G)$ et $P_i^-(G)$. L'ensemble de toutes les transitions potentiellement exécutables de l'entité P_i est noté : $P_i(G) = P_i^-(G) \cup P_i^+(G)$.

Lors de la construction d'un graphe de sauts, les transitions à exécuter simultanément, doivent être exécutables dans le même état global et doivent appartenir à des entités du protocole qui ne disposent pas de transitions potentiellement exécutables [8]. Un ensemble de telles transitions est appelé saut, noté *pleap* (proper leap). Donc, un saut est un ensemble non vide de transitions : $T \subseteq X(G)$, tel que pour toute transition $t, t^{'} \in T$, $act(t) \neq act(t^{'})$ si $t \neq t^{'}$ (i.e T contient au plus une transition exécutable de chaque entité). Les différents ensembles de T, construits à l'état global G, forment un ensemble de sauts noté $pleap(G)$.

Definition 1 (Calcul de l'ensemble *pleap* (sauts)[7]). *Soit G un état global. On note par $Wait(G)$, l'ensemble des indices des entités qui disposent d'une transition potentiellement exécutable à l'état G. Cet ensemble est définit par : $Wait(G) = \{i \in I / X_i(G) \Longrightarrow P_i(G) \neq \oslash\}$. L'ensemble des sauts $pleap(G)$ est donc calculé comme suite :*

$$pleap(G) = \begin{cases} \left\{\prod_{i \notin Wait(G)} X_i\,(G)\right\} & \text{si } Wait(G) \subset I \\ \{\{t\}\,/t \in X(G)\} & \text{sinon} \end{cases} \quad (1)$$

Avant de calculer l'ensemble $pleap(G)$, on doit calculer l'ensemble $Wait(G)$ pour identifier les entités qui disposent de transitions potentiellement exécutables. Si toutes les entités appartiennent à l'ensemble $Wait(G)$ ça veut dire qu'il n'y a pas de saut à faire à partir de G et dans ce cas la, on exécute chaque transition de $X(G)$ indépendamment des autres (transitions simples). Si l'ensemble $Wait(G)$ est inclut dans I, l'ensemble de sauts à exécuter à partir de G ($pleap(G)$) sera le produit cartésien des ensembles de transitions exécutables de chaque entité $Xi(G)$ i.e $pleap(G) = \{\prod_{i \notin Wait(G)}\{X_i\,(G)\}\}$.

Definition 2 (transition de saut [8]). *Une transition de saut est une relation binaire sur un ensemble d'états globaux, elle est notée : $\rightarrow^l$. Pour deux états globaux $G1 = (< s_i^{G1} >_{i \in I}, < c_{ij}^{G1} >_{(i,j) \in L})$ et $G2 = (< s_i^{G2} >_{i \in I}, < c_{ij}^{G2} >_{(i,j) \in L})$, $G1 \rightarrow^l G2$ existe, ssi il existe un ensemble de transitions T appartenant à l'ensemble de sauts $pleap(G1)$, tel que l'exécution simultanée des transitions de T, fait passer le protocole à l'état global $G2$.*
Soit $\rightarrow^{l*}$ la fermeture transitive et réflexive de $\rightarrow^l$. $G2$ est dit accessible par sauts depuis $G1$, ssi $G1 \rightarrow^{l*} G2$. Si $G1 = S0$ (état global initial du protocole), $G2$ est dit être accessible par saut. On dénote par L_Π l'ensemble de tous les états du protocole qui sont accessibles par sauts. Pour une séquence d'ensemble de sauts $\Omega = T1, T2, T3 \ldots, Tm$, $G1 \rightarrow^{l*} G2$ dénote l'existence des états globaux $Q0, Q1, Q2 \ldots, Qm$, tel que : $G1 = Q0 \rightarrow^{T1} Q1 \rightarrow^{T2} Q2 \ldots \rightarrow^{Tm} Qm = G2$.

Dans un graphe de sauts, les neouds sont des états globaux qui sont accessibles par des sauts. Le calcul des transitions de sauts (pleap) se fait au niveau de chaque état global, en commençant de l'état global initial du système.

3.2 Construction de chemins de sauts réversibles

Dans ce qui suit, nous donnons les définitions qui vont nous permettre de construire un graphe de sauts d'une manière réversible.

Definition 3 (état global réversible [5]). *Lorsque on parcourt un graphe d'accessibilité d'une manière réversible, l'ensemble des états construit par cette opération est appelé ensemble d'états globaux réversibles. Un état global réversible d'un protocole Π est noté $\underline{G} = (< s_i^{\overleftarrow{G}} >_{i \in I}, < c_{ij}^{\overleftarrow{G}} >_{(i,j) \in L})$ où $s_i^{\overleftarrow{G}}$ est l'état local de l'entité P_i et $c_{ij}^{\overleftarrow{G}}$ est la séquence de messages désirés sur le canal C_{ij}.*

L'opération de retour arrière peut être vue comme une procédure abstraite. A partir d'un état global on monte vers la racine, et dans ce cas-là, les transitions d'envoi de messages vont être traitées comme étant des transitions de réception de messages, et les transitions de réception de messages vont être traitées comme étant des transitions d'envoi de messages.

Transitions simultanément et réversiblement exécutables. Comme pour la construction des graphes de sauts, nous avons besoin de définir les notions de transitions réversiblement exécutables et de transitions potentiellement et réversiblement exécutables afin de calculer les transitions simultanément et réversiblement exécutables (sauts réversibles).

Transition réversiblement exécutable : transition $t = (s_i, \mu, s_i^{'}) \in \triangle_i$, est dite réversiblement exécutable à l'état global réversible $\underline{G}$, ssi $s_i^{'} = s_i^{'\underline{G}}$ ($s_i^{'\underline{G}}$ est l'état local de P_i à l'état global $\underline{G}$) , et t est sous une des deux formes suivantes :

1. t est une transition d'envoi de message tel que $\mu = -x$ avec $x \in M_i$ et $Front(c_{ij}^{\underline{G}}) = x$. ou,
2. t est une transition de réception de message et le contenu du canal c_{ji} est inférieure à K, i.e. $|C_{ji}|$ <K.

L'ensemble des transitions d'envoi et de réception de messages, qui sont réversiblement exécutables à l'état global réversible $\underline{G}$, sont notés $X_i^{R-}(\underline{G})$ et $X_i^{R+}(\underline{G})$ respectivement. L'union de ces deux ensembles est noté $X_i^R(\underline{G}) = X_i^{R-}(\underline{G}) \cup X_i^{R+}(\underline{G})$. L'ensemble de toutes les transitions du protocole, qui sont réversiblement exécutables à l'état $\underline{G}$, est noté : $X^R(\underline{G}) = \cup_{i \in I} X_i^R(\underline{G})$.

Transition potentiellement et réversiblement exécutable : Une transition de réception de message $t = (s_i, +y, s_i^{'})$, avec $y \in M_i$, est dite potentiellement et réversiblement exécutable à l'état global réversible $\underline{G}$, ssi $s_i^{'} = s_i^{'\underline{G}}$ et $|C_{ji}|$ =K ; K est la capacité des canaux de communication du système. Une transition d'envoi de message $t = (s_i, -x, s_i^{'})$, avec $x \in M_i$, est dite potentiellement et réversiblement exécutable à l'état global $\underline{G}$, ssi $s_i^{'} = s_i^{'\underline{G}}$ et $\underline{c_{ij}^{G}} = \varepsilon$. Les ensembles des transitions de réception de messages et des transitions d'envoi de messages de l'entité P_i, qui sont potentiellement et réversiblement exécutables à l'état global $\underline{G}$, sont notés $P_i^{R+}(\underline{G})$ et $P_i^{R-}(\underline{G})$ respectivement. On note $P_i^R(\underline{G}) = P_i^{R-}(\underline{G}) \cup P_i^{R+}(\underline{G})$.

Comme pour la construction d'un graphe de sauts, la construction d'un graphe de sauts réversibles se fait par le calcul des transitions simultanément et réversiblement exécutables, ces transitions doivent être exécutables dans le même état global réversible et doivent appartenir à des entités du protocole qui ne disposent pas de transitions potentiellement et réversiblement exécutables. Un ensemble de telles transitions est appelé sauts réversibles, noté $pleap^R$ (reverse proper leap). Donc, un saut réversible est un ensemble non vide de transitions : $T \subseteq X^R(\underline{G})$, tel que pour toute transition $t, t^{'} \in T$, $act(t) \neq act(t^{'})$ si $t \neq t^{'}$ (i.e T contient au

plus une transition réversiblement exécutable de chaque entité). Les différents ensembles T, construits à l'état global réversible $\underline{G}$, forment un ensemble de sauts réversibles noté $pleap^R(\underline{G})$.

Nous donnons dans ce qui suit la définition formelle de l'ensemble des sauts réversibles $pleap^R$.

Definition 4 (calcul de l'ensemble $pleap^R$). *Soit $\underline{G}$ un état global réversible. On définit l'ensemble des entités qui disposent d'une transition potentiellement et réversiblement exécutable à l'état $\underline{G}$ par : $Wait^R(\underline{G}) = \{i \in I / X_i^R(\underline{G}) \Longrightarrow P_i^R(\underline{G}) \neq \oslash\}$. L'ensemble $pleap^R(\underline{G})$ est calculé comme suite :*

$$pleap^R(G) = \begin{cases} \left\{\prod_{i \notin Wait^R(\underline{G})} X_i^R(\underline{G})\right\} & \text{si} \quad Wait^R(\underline{G}) \subset I \\ \left\{\{t\} / t \in X^R(\underline{G})\right\} & \text{sinon} \end{cases} \tag{2}$$

Avant de calculer l'ensemble $pleap^R(\underline{G})$, on doit calculer l'ensemble $Wait^R(\underline{G})$ pour identifier les entités qui disposent de transitions potentiellement et réversiblement exécutables. Si toutes les entités appartiennent à l'ensemble $Wait^R(\underline{G})$ ça veut dire qu'il n'y a pas de saut réversible à exécuter à partir de $\underline{G}$, et dans ce cas la, on exécute chaque transition de $X^R(\underline{G})$ indépendamment des autres (simple transition). Si l'ensemble $Wait^R(\underline{G})$ est inclut dans I, l'ensemble de sauts réversibles $pleap^R(\underline{G})$ sera le produit cartésien des ensembles de transitions réversiblement exécutables de chaque entité($X_i^R(\underline{G})$), i.e $pleap^R(\underline{G}) = \{\prod_{i \notin Wait^R(\underline{G})}\{X_i^R(\underline{G})\}\}$.

Definition 5 (transition de saut réversible). *Une transition de saut réversible, est une relation binaire $\rightarrow^R$ sur un ensemble d'états globaux réversibles. Pour deux états globaux réversibles $\underline{G1} = (< s_i^{G1} >_{i \in I}, < c_{ij}^{G1} >_{(i,j) \in L})$ et $\underline{G2} = (< s_i^{G2} >_{i \in I}, < c_{ij}^{G2} >_{(i,j) \in L})$, $\underline{G1} \rightarrow^R \underline{G2}$ existe, ssi il existe un ensemble de transitions réversiblement exécutables $T \in pleap^R(\underline{G1})$, tel que l'exécution des transitions de T à partir de $\underline{G1}$ donne l'état global $\underline{G2}$. La relation de transition $\underline{G1} \rightarrow^R \underline{G2}$ signifie que l'état global $\underline{G2}$ est obtenu d'une manière réversible directe depuis l'état global $\underline{G1}$. Soit $\rightarrow^{R*}$ la fermeture transitive et réflexive de $\rightarrow^R$, on dit que l'état $\underline{G2}$ est accessible par saut réversible de puis $\underline{G1}$, ssi $\underline{G1} \rightarrow^{R*} \underline{G2}$. Le chemin qui relie $\underline{G1}$ vers $\underline{G2}$ est alors appelé : chemin de sauts réversibles de $\underline{G1}$ vers $\underline{G2}$.*

3.3 Traitement des erreurs d'interblocage

Dans cette section, nous expliquons l'utilisation des sauts réversibles pour la détection des erreurs d'interblocage.

Definition 6 (état suspect d'interblocage [5]). *Un état global $G = (< s_i^G >_{i \in I}, < c_{ij}^G >_{(i,j) \in L})$ est dit un possible état d'interblocage ssi, $\forall (i,j) \in L$: $c_{ij}^G = \varepsilon$ et s_i^G est un état de réception de message pour tout $i \in I$.*

Definition 7 (état d'interblocage [8]). *Un état global $G = (< s_i^G >_{i \in I}, < c_{ij}^G >_{(i,j) \in L})$ est dit état d'interblocage ssi :*

1. $\forall (i,j) \in L : c_{ij}^G = \varepsilon$ *et* s_i^G *est un état de réception de message pour tout* $i \in I$;
2. *L'état global* G *est accessible par sauts à partir de* $G0$.

Un état global est dit état d'interblocage lorsque tous les canaux de communications dans cet état sont vides et toutes les entités du protocole sont dans un état de réception de messages.

Nous donnons ci-dessous, le théorème principal de notre proposition, qui nous permettra d'utiliser les chemins de sauts réversibles afin de détecter les erreurs logiques de type interblocage.

Theorem 1. *Soit* $Pi = (P, L)$ *un protocole et soit* GS *un état global suspect d'interblocage.*
L'état suspect d'inter blocage GS*, est confirmé comme état d'inter blocage ssi, l'état global initial du protocole* $G0$ *est accessible par un chemin de sauts réversibles à partir de l'état suspect* GS.

On se basant sur le théorème 1, la détection des états d'interblocage, peut être faite par une vérification de l'accessibilité par sauts réversibles de l'état initial du système en partant d'un état suspect.

3.4 Algorithme

Dans cette section, nous donnons l'algorithme de notre approche de validation. Pour des raisons de simplifications, nous traitons des protocoles de communication entre deux entités seulement. Notre algorithme peut être facilement généralisé sur une communication entre plusieurs entités.

Détection des états suspects (Phase1).
Les états suspects d'interblocage sont obtenus par une combinaison, des états locaux de réception de messages, des différentes entités communicantes du protocole (Définition 6). Cette phase est implémentée par l'Algorithme 1 du calcul d'états suspects.

Le test de la ligne 4 de l'algorithme sert à écarter les états locaux qui sont finaux car un état global final ne peur être état d'interblocage.

Confirmation des états d'interblocage (Phase2).
Afin de confirmer l'existence d'une erreur dans un état suspect GS, l'algorithme effectue une recherche d'un chemin de sauts réversibles vers l'état initial du protocole.
Cette phase est donnée sur l'Algorithme 2. L'ensemble *Explored* est utilisé pour marquer les états globaux qui sont déjà traités (vérifiés). Chaque nouvel état global n'appartenant pas à *Explored*, sera mis dans *Open* afin de le traiter dans la prochaine itération (ligne 20-21-22). A partir de chaque état global $G^{'}$ (non traité), nous calculons tous les ensembles des transitions simultanément exécutable (T) de cet état global, par l'utilisation de la définition 4 (ligne 18).

Algorithm 1 Phase de détection des états suspects

Require: Entrées :protocole $\Pi(\mathrm{P1},\mathrm{P2})$;
P_i=(S_i,$S0_i$,Ef_i,M_i,$\triangle_i$) i∈{1,2} ;
Sortie :*Suspect* ;//ensemble des états suspects

1: $Suspect$=$\emptyset$;
2: **pour** chaque état de réception de message s_1 dans $S1$ **faire**
3: **pour** chaque état de réception de message s_2 dans $S2$ **faire**
4: **si** s_1 et s_2 sont des états non finaux **alors**
5: $GS = (s_1, s_2, \varepsilon, \varepsilon)$;//générer un état suspect
6: $Suspect = Suspect \cup \{GS\}$;
7: **finsi**
8: **fin pour**
9: **fin pour**

Algorithm 2 Phase de confirmation des états d'interblocage

Require: Entrées :protocole $\Pi(\mathrm{P1},\mathrm{P2})$;
P_i=(S_i,$S0_i$,Ef_i,M_i,$\triangle_i$) i∈{1,2} ;
Etat global suspect GS : $(s_1, s_2, \varepsilon, \varepsilon)//s_1 \in S1, s_2 \in S2$
Sortie :Confirmation ou non de l'erreur dans l'état suspect GS ;

1: $Explored = \emptyset$;
2: $Open = \{GS\}$;
3: **tantque** $Open \neq \emptyset$ **faire**
4: Prendre un état global G de $Open$;
5: $Open = Open - \{G\}$;
6: $Explored = Explored \cup \{G\}$;
7: **si** $G = S0$ **alors**
8: **si** $G.s_1$ est complexe ET $G.s_2$ est complexe **alors**
9: retourne : GS est confirmé comme erreur d'interblocage complexe ;
10: **sinon**
11: **si** $G.s_1$ est complexe OU $G.s_2$ est complexe **alors**
12: retourne : GS est confirmé comme erreur d'interblocage hybride ;
13: **sinon**
14: retourne : GS est confirmé comme erreur d'interblocage simple ;
15: **finsi**
16: **finsi**
17: **sinon**
18: calculer $pleap^R(G)$; // calcul des sauts réversibles à partir de G
19: **pour tout** T dans $pleap^R(G)$ **faire**
20: Générer un état $G^{'}$ par l'exécution simultané des transitions de T ;
21: **si** $G^{'} \notin Explored$ **alors**
22: $Open = Open \cup G^{'}$
23: **finsi**
24: **fin pour**
25: **finsi**
26: **fin tantque**
27: retourne :GS n'est pas une erreur de protocole ;

Pour chaque T obtenu et qui est un saut réversible, nous générons un nouvel état global. L'algorithme continue son exécution jusqu'à ce que soit l'état initial est atteint, et dans ce cas la l'erreur logique est confirmée dans l'état suspect GS, soit il ne reste plus d'état à explorer et donc l'erreur logique est écartée. Les lignes de code de 7 à 16 permettent d'identifier le type de l'erreur détectée (simple, hybride ou complexe), selon la nature des états locaux composant l'état de l'erreur.

4 Résultas d'expérimentation

Pour montrer l'efficacité de notre technique, nous avons choisi de la comparer avec une technique de la même famille, c'est-à-dire une technique qui utilise le principe de retour arrière. La technique choisie est l'analyse d'accessibilité réversible (Reverse Reachability analysis) RRA. Cette technique a prouvée son efficacité par rapport à plusieurs d'autres techniques de validation. La comparaison se fait sur la taille du graphe d'accessibilité généré afin de valider le protocole. La taille d'un graphe d'accessibilité est mesurée par rapport au nombre d'états globaux générés et par rapport au nombre de transitions exécutées pour sa construction.
Nous avons fait notre comparaison sur le même ensemble d'exemples utilisé pour valider la technique RRA. Le Tableau 1 montre la réduction apportée par notre technique sur l'ensemble des exemples de comparaison selon la variation en taille des canaux de communication.

Capacité des canaux C12=C21	Réduction sur la taille du graphe d'accessibilité : RLRA/RRA	
	Nombre d'états explorés	Nombre de transitions exécutées
1	11,11%	20,54%
2	21,42%	32,20%
3	4%	12,68%
4	2,17%	6,06%
5	6,42%	7,38%

Tab. 1. Réduction apportée par la technique RLRA par rapport à la technique RRA

5 Conclusion

Dans cet article, nous avons proposé une technique de validation de protocoles basée sur une approche de validation avec retour arrière. Pour rendre l'analyse plus efficace, nous avons optimisé l'opération de construction des chemins de retour arrière. Cette optimisation est obtenue par l'utilisation de la technique de construction des graphes de sauts(LRA). L'utilisation de LRA nous a permet d'éviter de parcourir des états qui ne sont pas utiles dans l'analyse, ce qui nous a permet de réduire considérablement la taille du graphe d'accessibilité.

Références

1. Choi, T.Y. : A structured approach to the analysis and design of finite state protocols. Ph.D.Thesis, School of Electrical Engineering, Georgia Institute of Technology, 1983.
2. Gouda, M.G., and Yu, Y.T. : Protocol validation by maximal progress state exploration. in Proceedings of ACM SIGCOMM, pp. 68-75, 1983.
3. Lin, F.J., Chu, P.M. and Liu, M.T. : Protocol verification using reachability analysis : the state space explosion problem and relief strategies, Computer communications Review, volume 17, no.5, pp. 126-143, 1987.
4. Peng, w. and Purushothaman, S. : Data flow analysis of communicating finite state machines. ACM Transactions on Programming Languages and Systems, Vol. 13, No. 3, pp.399-442, July 1991.
5. Hung, Y.C., and Chen, G.H., "Reverse reachability analysis : a new technique for deadlock detection on communicating finite state machines", Softwar Practice and Experience, September, volume 23(9), pp.88-93, 1993.
6. Ozdemir, K. : Verifying the safety properties of concurrent systems via simultaneous reachability, Ph.D. Thesis, Department of CSI, University of Ottawa, 1995.
7. Hans, V.S., and Hasan, U. ; A uniform approach to tackle state explosion in verifying progress properties for networks of CFSMs*. Department of Computer Science, University of Ottawa, TR-96-13, November 1996.
8. Ozdemir, K. and Ural, H. : Protocol validation by simultaneous reachability analysis, Computer Communications, 20(9) : 772-788, 1997.
9. Tari, Z. and Arora, P. : A Communication Protocol Validation Approach based on Partial Exploration of Complex State Machines ,ICDCIT, 2007.