Analyse quantitative paramétrée d'automates temporisés probabilistes

Najla Chamseddine

THÈSE

présentée à l'École Normale Supérieure de Cachan

pour obtenir le grade de

Docteur de l'École Normale Supérieure de Cachan

par : Najla Chamseddine

Spécialité : Informatique.

Analyse quantitative paramétrée d'automates temporisés probabilistes

Jury :

Béatrice Bérard	Rapportrice
Laurent Fribourg	Co-directeur de thèse
Claudine Picaronny	Co-directrice de thèse
Jeremy Sproston	Examinateur
François Vernadat	Rapporteur.

Sommaire

Introduction

Contexte

De nombreux systèmes informatiques existant à différents niveaux de complexité et d'utilisation, font apparaître certaines défaillances ou dysfonctionnements. Dans certains cas, ces erreurs de fonctionnement n'ont pas d'impact grave sur l'environnement qui les entourent. D'autres peuvent par contre avoir dessus des répercussions néfastes. Ces systèmes sont dits *critiques*.

Par exemple, certains accidents aériens qui mènent le plus souvent à des pertes humaines peuvent être causés par de telles anomalies. Une défaillance dans les systèmes embarqués comme les sondes de vitesse ou les systèmes détectant la distance de l'avion du sol ou dans une mauvaise interaction homme-machine peut entraîner une catastrophe aérienne. Dans le cas d'accident en plein vol, il faut aussi tenir compte du facteur environnant car des orages actifs peuvent déstabiliser le fonctionnement normal de l'avion.

Il est ainsi nécessaire de vérifier de tels systèmes *formellement* en essayant de penser à tous les scénarios possibles que peut rencontrer l'avion et traiter ceux qui pourraient mener à un dysfonctionnement en décelant les événements qui ne devraient pas se produire a priori.

Intervient ainsi à ce niveau, la notion de *vérification* de systèmes. Etant donné un système à étudier $\mathcal{S}$, on choisit un formalisme adapté pour vérifier un ensemble $\mathcal{P}$ de propriétés.

Le formalisme choisi permet d'abstraire le système en un modèle $\mathcal{M}$. Dans l'exemple des systèmes aériens, il est intéressant de vérifier la propriété *Jamais un crash ne peut survenir*. Des données probabilistes (par exemple, la probabilité qu'un réacteur soit en panne) peuvent intervenir et mener à la vérification de propriétés probabilistes comme *La probabilité qu'un crash ait lieu est inférieure à* 10^{-4}.

La vérification se présente sous trois approches différentes : le *model checking*, la *preuve assistée* et le *test*.

La vérification par model checking permet de vérifier des propriétés exprimées sous forme de propositions mathématiques, par exemple comme des formules de *logique temporelle* sur le modèle $\mathcal{M}$ du système à étudier. On cherche ainsi à savoir si un modèle $\mathcal{M}$ vérifie une formule ϕ, noté $\mathcal{M} \models \phi$. Le modèle est décrit par des configurations qui décrivent son état à un instant donné et des transitions entre ces configurations. Une exécution est une suite de telles transitions. Et pour décrire son évolution dans le temps, une donnée temporelle dans $\mathbb{R}$ ou $\mathbb{N}$ est souvent nécessaire pour exprimer les délais et les temps d'attente. Les propriétés à vérifier portent sur les états ou sur les transitions des exécutions (ou chemins) du système. Plusieurs

formalismes sont proposés pour la description de systèmes : les *Automates Temporisés*, les *Réseaux de Petri*, etc ...

Si une donnée probabiliste est intégrée dans la modélisation du système, nous obtenons des systèmes temporisés et probabilistes.

La vérification par preuve assistée utilise des techniques de déduction afin de donner des preuves formelles de propriétés.

Enfin, le test s'opère en exprimant des scénarios particuliers soumis au système et en comparant ensuite le résultat en sortie au résultat souhaité.

Nous nous intéressons dans notre travail au model checking. Les propriétés peuvent être particulières au système selon la spécification définie pour ce dernier, ou typiques telles que la vivacité, la sûreté ou l'équité.

La sûreté permet de vérifier que *quelque chose de mauvais n'arrive jamais* tandis que la vivacité indique que *quelque chose de bon est toujours possible*. L'équité est soit forte soit faible. L'équité forte permet de vérifier si un événement qui peut avoir lieu infiniment souvent a effectivement lieu infiniment souvent. Dans l'équité faible, on vérifie si un événement qui peut toujours avoir lieu à partir d'un certain moment a lieu infiniment souvent.

Ces propriétés sont vérifiées sur des systèmes informatiques tels que les protocoles cryptographiques ou probabilistes, les systèmes réactifs, distribués ou temps-réel, les systèmes probabilistes ... On distingue deux types de propriétés : qualitatives et quantitatives. On vérifie au travers de propriétés qualitatives si un événement a lieu ou non dans les différentes exécutions possibles de $\mathcal{S}$. On cherche par les propriétés quantitatives à déduire par des méthodes de calcul appliquées au modèle $\mathcal{M}$ des informations temporelles (ou probabilistes) portant sur le système.

Problématique

Nous nous intéressons dans ce travail à des propriétés quantitatives d'un type particulier, relatives au temps mis par le système pour atteindre un ensemble particulier E d'états. Ces informations peuvent être cruciales par exemple pour borner les phases de stabilisation d'un système ou les temps de réponses (exemple : une alarme se produira au plus tard 10 secondes après la survenue d'un signal d'erreur).

La vérification de telles propriétés, dites de performance, repose sur le calcul de *temps de convergence* vers un ensemble d'états E accessibles dans un système temporisé. Il faut donc tout d'abord vérifier qualitativement que tout état de E est accessible puis calculer le temps qu'il faut pour atteindre l'ensemble E. Les temps d'attente et les délais sont exprimés dans $\mathbb{N}$ ou dans $\mathbb{R}$ mais peuvent aussi être représentés par des paramètres dans le modèle $\mathcal{M}$. Un seul modèle représenterait alors un ensemble de modèles correspondant aux valeurs des paramètres apparaissant dans $\mathcal{M}$. Ces valeurs doivent néanmoins satisfaire les contraintes propres au système et adaptées à sa spécification physique.

Les principales difficultés de cette approche résident dans les aspects non déterministes des systèmes : nous ne pouvons pas prévoir avec certitude l'état suivant dans lequel se trouvera le système. Cet aspect lié à l'incertitude peut être modélisé de différentes façons : non déterministe au niveau des actions et/ou ajout de probabilités.

La présence de non déterminisme dans le choix des événements peut particulièrement nous amener à rechercher un temps minimal ou maximal de convergence vers cet état. En effet,

le temps de convergence varie en fonction de l'événement que l'on choisit d'exécuter à partir d'un état courant. Cet événement peut ainsi imposer au système un délai d'attente dans l'état courant supérieur ou inférieur à ceux proposés par les autres événements possibles.

Dans le cas où le système comporte un facteur probabiliste dans le choix des événements, nous parlerons d'une *espérance de temps de convergence* ou *temps moyen de convergence* vers un ensemble d'états accessibles E. Dans chaque état du système probabiliste, les probabilités sont représentés par une ou plusieurs distributions. Le non déterminisme intervient dans le choix de la distribution et le temps varie alors selon la distribution choisie.

Le travail que nous présenterons porte sur le calcul du temps maximal d'atteinte d'un état cible dans ce type de système. Ce *temps moyen de convergence maximal* parmi tous les temps moyens de convergence est appelé *pire temps moyen de convergence* vers un état cible accessible.

Aspect non déterministe

L'environnement joue un rôle important dans l'abstraction du système par un modèle $\mathcal{M}$. Un facteur non déterministe lié à l'état potentiel de l'environnement doit donc être pris en compte dans la modélisation $\mathcal{M}$ de $\mathcal{S}$.

Prenons l'exemple d'un système distribué $\mathcal{S}$ composé de sous systèmes qui interagissent entre eux. L'environnement de chaque composant est une partie ou l'ensemble des autres systèmes composant $\mathcal{S}$. A un instant donné, le système doit choisir d'exécuter un événement parmi tous les événements possibles que peut procurer l'ensemble ou une partie des composants. Le non déterminisme qui intervient à ce niveau est représenté dans le modèle $\mathcal{M}$ du système $\mathcal{S}$. Les propriétés d'équité peuvent être vérifiées dans ce type de système pour montrer que le choix non déterministe ne se fait pas toujours sur le même événement et qu'un composant prêt à exécuter une action à plusieurs instants du processus finira par le faire ultérieurement.

Aspect probabiliste

Dans un système probabiliste, la possibilité qu'un événement se produise en partant d'un état du système est quantifiée par une probabilité. Une ou plusieurs distributions sont alors associées à cet état du système. Le non déterminisme est ici présent dans le choix de la distribution qui va ensuite permettre l'exécution d'un événement avec une certaine probabilité.

Dans le cas probabiliste, une vérification qualitative permet de savoir si une propriété du système se produit avec une probabilité égale à 0, à 1 ou est strictement comprise entre 0 et 1. On vérifie par exemple la propriété de convergence : *tous les chemins, sauf un ensemble de probabilité nulle, satisfont la propriété* ϕ où ϕ est une propriété de vivacité comme *en partant d'un état quelconque du système on atteint un ensemble E d'états cibles.*

Une vérification quantitative concerne la probabilité exacte avec laquelle le système satisfait la propriété ϕ. Elle peut aussi porter sur le calcul de temps de *convergence* vers un ensemble d'états cibles.

L'aspect probabiliste est important pour la modélisation de certains systèmes informatiques. Ainsi, dans le cas des protocoles de communication le facteur probabiliste apparaît au niveau des chances d'envoi d'un message complet, du choix du temps d'attente que doivent avoir les émetteurs avant de réémettre leurs messages ou des chances de perte de données lors

de la transmission.

Les principaux formalismes existants pour la modélisation de systèmes probabilistes sont les *Processus de Décision Markovien* et les *Chaînes de Markov*. Un processus de décision markovien attribue à chaque état du système un ensemble de distributions. Les chaînes de Markov sont un cas particulier des processus de décision markovien car chaque état possède une unique distribution. On appelle *automate à coûts positifs* une chaîne de Markov à laquelle on attribue des coûts positifs, dans $\mathbb{R}$ ou dans $\mathbb{N}$, à ses transitions.

Contribution

Modèles temporisés probabilistes

Les *Automates Temporisés Probabilistes* constituent le formalisme sur lequel se basera la modélisation des systèmes que nous étudions. Ils sont en effet une extension probabiliste des Automates Temporisés. Le temps est représenté de manière dense dans $\mathbb{R}$ ou discrète dans $\mathbb{N}$. Si le temps est représenté par des paramètres dont chacun peut prendre un ensemble de valeurs, nous parlerons d'*Automate Temporisé Probabiliste Paramétré*. Un automate temporisé probabiliste paramétré est donc un ensemble d'automates temporisés probabiblistes.

A chaque état d'un automate temporisé probabiliste est associé un ensemble fini de distributions. Le non déterminisme dans les automates temporisés probabilistes se présente sous deux formes : l'une porte sur les délais d'attente possibles dans chaque état du système et l'autre sur le choix de la distribution dans un état donné. Le temps est en fait une autre source de non déterminisme. Si l'on considère par exemple un émetteur qui doit envoyer son message entre 4 et 6 unités de temps après que le canal de transmission ne soit libre, on aura un temps d'attente compris entre 4 et 6 unités de temps. Dans le cas continu, on considère toutes les possibilités comprises dans l'intervalle [4,6] alors que dans le cas discret nous avons les trois possibilités 4, 5 et 6 unités de temps.

Un automate temporisé probabiliste peut être vu comme un processus de décision markovien où l'écoulement du temps est représenté par des transitions successives de probabilité 1. Cette transformation ne peut cependant pas aboutir dans $\mathbb{R}$ si l'on veut représenter tous les délais d'attente possibles dans les états du système.

Pour cela, on peut faire appel à la technique de discrétisation du temps dans l'automate temporisé probabiliste pour obtenir un processus de décision markovien en considérant l'écoulement du temps dans $\mathbb{N}$ et non dans $\mathbb{R}$. Ceci permet certes de réduire le nombre de possiblités de temps d'attente mais une explosion du nombre d'états lors de la transformation peut survenir. Dans le cas où les temps sont représentés par des paramètres, la transformation dépendra en plus de la taille des valeurs attribuées aux paramètres.

Réduction du modèle

Les travaux qui portent sur le calcul de temps moyens de convergence minimal ou maximal dans un automate temporisé probabiliste utilisent la technique de discrétisation du temps. Cette technique peut cependant entraîner une explosion du nombre d'états du processus de décision markovien nécessaire pour calculer les temps moyen de convergence minimal ou maximal en utilisant des techniques de programmation linéaire.

Plusieurs techniques de vérification font appel à des réductions précises du modèle suivant des relations d'ordre en fonction de la propriété à vérifier, comme par exemple la *bisimulation*. Dans notre cas, étant donné un automate temporisé probabiliste, nous voulons calculer le pire temps moyen de convergence vers un unique état accessible final (ou temps moyen de convergence maximal).

Le principal problème qui se pose lors du calcul du pire temps moyen de convergence en discrétisant le temps est le maintien des aspects non déterministes, qu'ils soient temporels ou probabilistes. S'ajoute à cela la taille des valeurs attribuées aux paramètres dans le cas des automates temporisés probabilistes paramétrés.

Ainsi, tout en voulant calculer le pire temps moyen de convergence, nous proposons une technique adaptée à ce type de vérification pour une sous classe de systèmes. On est en effet amené dans ce type de calcul à se focaliser sur les aspects non déterministes qui seuls font varier le temps moyen de convergence à cause des choix effectués dans les états. Pour obtenir un temps moyen maximal, le choix non déterministe dans un état donné du système pourrait porter sur le temps d'attente le plus long et la distribution qui l'y autorise. On fixe donc un temps d'attente et une distribution dans le but de capturer un temps moyen de convergence maximal vers l'état cible, ce qui ôterait toutes les formes de non déterminisme de l'automate temporisé probabiliste initial.

Sous de bonnes hypothèses, on peut capturer le comportement du système qui correspond au temps moyen de convergence maximal vers l'état cible. On lui applique alors des techniques de calcul moins lourdes que les techniques de programmation linéaire et inspirées de celles utilisées sur les chaînes de Markov.

Plan

Nous proposons dans cette thèse une sous classe d'automates temporisés probabilistes paramétrés à partir duquel nous pouvons calculer le temps moyen de convergence maximal en utilisant des techniques connues de calcul sur les chaînes de Markov.

Nous explicitons formellement dans le **chapitre 1** les modèles probabilistes que nous avons cités ci-dessus. Nous présentons tout d'abord dans le **chapitre 2** quelques propriétés nécessaires pour définir la classe d'automates temporisés probabilistes qui sont :

- le *non blocage* et les comportements *non zenon* et *fortement non zenon* dans un automate temporisé probabiliste.
- le *déterminisme* d'actions dans un automate temporisé probabiliste.
- Un unique état *final* et *absorbant*.

A partir de ces propriétés prises pour hypothèses, nous définissons les Automates Temporisés Probabilistes Paramétrés Déterminés. Il faut en effet que l'automate soit *non bloquant* et *fortement non zenon* pour définir le temps moyen de convergence vers son unique état final et absorbant. On vérifie ainsi que l'on peut atteindre l'état final et absorbant de tout état de l'automate.

Enfin, le **chapitre 3** se présente en deux parties. Nous développons dans la première partie un algorithme qui transforme l'automate temporisé probabiliste paramétré déterminé en un graphe semblable aux chaînes de Markov à coûts qu'on appellera *graphe des macro-steps*. Dans un deuxième temps, on utilisera les méthodes qui existent déjà pour les chaînes

de Markov à coûts pour le calcul du temps moyen de convergence dans ce graphe. Le résultat obtenu est paramétré. Nous démontrons par ailleurs que le temps moyen de convergence dans l'automate temporisé probabiliste paramétré déterminé est égal au temps moyen de convergence dans le graphe des macro-steps. Ceci permet de déduire finalement le pire temps moyen de convergence dans l'automate d'origine, duquel nous avons obtenu la forme réduite purement déterministe.

Résumé

Nous considérons une sous classe d'automates temporisés probabilistes où les contraintes temporelles au niveau des gardes et des invariants sont exprimées par des paramètres. Cette sous classe est appelée la classe des automates Temporisés Probabilistes Paramétrés Semi Déterminés (ATPP Semi Déterminés). Cette classe d'automates se définit en particulier par l'attribution d'une unique distribution à chaque état et par des gardes de la forme $x \leq a$ où a est un paramètre ou un entier naturel. Nous imposons de plus deux propriétés sur ces automates qui sont celles de non blocage et fortement non zenon.

Notre travail vise à calculer le temps moyen maximal de convergence vers un état dit absorbant q_{end} dans ce type d'automates. L'unique méthode traitant déjà ce type de problème fait appel à la discrétisation du temps et à l'application de techniques de programmation linéaire. Elle est cependant exponentielle car elle dépend du nombre d'horloges et de la plus grande constante à laquelle sont comparés les horloges, lors de la discrétisation. Le graphe résultant peut être de taille exponentielle.

Pour tout ATPP Semi Déterminé, on définit un automate totalement déterministe, appelé ATPP Déterminé, en remplaçant toute garde de la forme $x \leq a$ par une garde de la forme $x = a$. Le temps d'attente en chaque état est ainsi fixé par la valuation de l'état initial qui remet toutes les horloges à zéro. Nous démontrons que le temps moyen de convergence vers q_{end} dans l'ATPP Déterminé est égal au temps moyen maximal de convergence dans l'ATPP Semi Déterminé dont il découle.

Pour calculer le temps moyen de convergence vers q_{end} nous construisons à partir de l'ATPP Déterminé un graphe appelé *graphe des macro-steps* qui contient de façon concise l'information nécessaire au calcul du coût moyen de convergence vers q_{end}. Ce graphe est de taille polynomiale et se construit en temps polynomial. Le calcul du temps moyen de convergence dans le graphe des macro-steps est solution d'un système linéaire, comme dans le cas des chaînes de Markov avec coûts. On résout ce système linéaire en temps polynomial, ce qui permet d'obtenir finalement le temps moyen maximal de convergence vers q_{end} dans l'ATPP Semi Déterminé.

Nous appliquons enfin cette méthode à certains protocoles de communication, notamment BRP (Bounded Retransmission Protocol) et CSMA/CD (Carrier Sense Multiple Access with Collision Detection).

Chapitre 1

Modèles

1.1 Les automates temporisés

1.1.1 Définitions

Si E est un ensemble, nous notons E^* l'ensemble des suites finies d'éléments de E. Nous considérons comme domaine temporel $\mathbb{T}$ l'ensemble $\mathbb{Q}^+$ des rationnels positifs ou nul ou l'ensemble des réels $\mathbb{R}^+$ positifs ou nuls et Σ un ensemble fini d'actions. Une *suite temporelle* sur $\mathbb{T}$ est une suite finie et croissante $\tau = (t_i)_{1\leq i\leq p} \in \mathbb{T}^*$. Un *mot temporisé* $\omega = (a_i, t_i)_{1\leq i\leq p}$ est un élément de $(\Sigma \times \mathbb{T})^*$ qui peut aussi être défini par une paire $w = (\sigma, \tau)$, avec $\sigma = (a_i)_{1\leq i\leq p}$ un mot dans Σ^* et $\tau = (t_i)_{1\leq i\leq p}$ une séquence temporelle dans $\mathbb{T}^*$.

Valuation temporelle. Nous considérons un ensemble fini $\mathcal{X}$ de variables, appelées horloges. Une *valuation d'horloge* sur l'ensemble $\mathcal{X}$ est une application $\nu : \mathcal{X} \to \mathbb{T}$ qui associe à chaque horloge une valeur dans $\mathbb{T}$. L'ensemble de toutes les valuations sur $\mathcal{X}$ est noté $\mathbb{T}^{\mathcal{X}}$. Soit $t \in \mathbb{T}$, la valuation $\nu + t$ est définie par $(\nu + t)(x) = \nu(x) + t$, $\forall x \in \mathcal{X}$. Si l'on numérote les horloges ($|\mathcal{X}| = n$ et $\mathcal{X} = \{x_1, \cdots, x_n\}$), nous utilisons aussi la notation $(\alpha_i)_{1\leq i\leq n}$ pour toute valuation ν tel que $\nu(x_i) = \alpha_i$. Pour un sous-ensemble X de $\mathcal{X}$, nous notons $[X := 0]\nu$ la valuation telle que pour tout $x \in X$, $([X := 0]\nu)(x) = 0$ et pour tout $x \in \mathcal{X} \setminus X$, $([X := 0]\nu)(x) = \nu(x)$.

Contrainte temporelle. Soit $\mathcal{X}$ un ensemble d'horloges. On introduit un ensemble de contraintes temporelles sur $\mathcal{X}$. Cet ensemble est noté $\mathcal{C}(\mathcal{X})$ et est défini par la grammaire suivante :

$$\phi ::= x \sim c \mid \phi \wedge \phi \mid true$$

où $x, y \in \mathcal{X}$, $c \in \mathbb{Z}$, $\sim \in \{<, \leq, =, \geq, >\}$

On écrit $\nu \models \phi$ lorsque la valuation temporelle ν satisfait la contrainte temporelle ϕ.

Définition 1.1. *Une contrainte temporelle k-bornée est une contrainte temporelle dans laquelle interviennent uniquement des constantes entre $-k$ et $+k$.*

Définition 1.2. (*Automates temporisés.*) *Un automate temporisé [2, 10] sur* $\mathbb{T}$ *est un uplet* $\mathcal{A} = (\Sigma, Q, T, I, F, \mathcal{X}, inv)$*, où* Σ *est un alphabet fini d'actions,* Q *un ensemble fini d'états,* $\mathcal{X}$ *un ensemble fini d'horloges,* $T \subseteq Q \times [\mathcal{C}(\mathcal{X}) \times \Sigma \times 2^{\mathcal{X}}] \times Q$ *est un ensemble fini de transitions, la fonction* $inv : Q \to \mathcal{C}(\mathcal{X})$ *associe à chaque état dans* Q *une conjonction de contraintes temporelles,* $I \subseteq Q$ *est un sous ensemble d'états initiaux et* $F \subseteq Q$ *est un sous ensemble d'états finaux.*

Pour toute transition $e = (q, \phi, a, X, q')$, nous avons donc :
- un état source q aussi noté $source(e)$ et un état cible q' aussi noté $cible(e)$.
- Une conjonction de contraintes temporelles sur $\mathcal{X}$ ϕ appelée garde de la transition e et aussi notée $garde(e)$, qui définit un polyèdre convexe sur $\mathcal{X}$.
- l'étiquette a aussi notée $label(e)$.
- l'ensemble des horloges X remises à zéro noté $reset(e)$

Pour tout $q \in Q$, $inv(q)$ est appelé l'*invariant* de q.
Pour un état q de l'automate temporisé $\mathcal{A}$, on écrit $in(q)$ l'ensemble des transitions de cible q (de la forme $(-,-,-,-,q)$) et $out(q)$ l'ensemble des transitions de source q (de la forme $(q,-,-,-,-)$).
Chaque état $q \in Q$ d'un automate temporisé $\mathcal{A}$ est aussi muni d'une conjonction de contraintes temporelles sur $\mathcal{X}$ qui forme un polyèdre convexe sur $\mathcal{X}$ et qui représente l'invariant de q.

Remarque 1.1. *Nous pouvons aussi ajouter des contraintes diagonales de la fome* $x - y \sim c$*. Les automates temporisés sans contraintes diagonales ont cependant la même expressivité que les automates temporisés avec contraintes diagonales [2, 5]. On peut en effet construire à partir d'un automate comportant des contraintes diagonales un autre automate qui lui est équivalent et sans contraintes diagonales. Nous considérons ainsi que des automates temporisés sans contraintes diagonales.*

Un *chemin* fini dans $\mathcal{A}$ est une suite finie de transitions successives :

$$P = q_0 \xrightarrow{\phi_1, a_1, X_1} q_1 \cdots q_{p-1} \xrightarrow{\phi_p, a_p, X_p} q_p$$

où $(q_{i-1}, \phi_i, a_i, X_i, q_i) \in T$, pour tout $1 \leq i \leq p$. Le chemin est dit *acceptant* si l'état initial $q_0 \in I$ et l'état final $q_p \in F$.

Une *exécution* de l'automate sur le chemin P est une suite de la forme :

$$\omega = \langle q_0, \nu_0 \rangle \xrightarrow[t_1]{\phi_1, a_1, X_1} \langle q_1, \nu_1 \rangle \cdots \xrightarrow[t_p]{\phi_p, a_p, X_p} \langle q_p, \nu_p \rangle$$

où $q_0 \in I$, $\tau = (t_i)_{1 \leq i \leq p}$ est une séquence temporelle, $\nu_0(x) = 0$, $\forall x \in \mathcal{X}$ et $(\nu_i)_{1 \leq i \leq p}$ sont les valuations temporelles.

Dans une exécution, un état est une configuration de l'automate à un instant donné. Pour cela, l'état $\langle q, \nu \rangle$ appartient à $Q \times \mathbb{T}^{\mathcal{X}}$ pour traduire l'état de contrôle q dans lequel se trouve l'automate et les valeurs des horloges à un instant donné données par la valuation ν. En

reprenant l'exemple de l'exécution ω, une transition $\langle q_{i-1}, \nu_{i-1}\rangle \xrightarrow[t_i]{\phi_i, a_i, X_i} \langle q_i, \nu_i\rangle$ représente un écoulement de temps de durée t_i puis un passage de l'état q_{i-1} à l'état q_i. Ceci se décrit par les deux transitions successives suivantes :

1. $\langle q_{i-1}, \nu_{i-1}\rangle \xrightarrow{t_i} \langle q_{i-1}, \nu_{i-1} + t_i\rangle$ qui correspond à l'écoulement du temps dans l'état q_{i-1},
2. puis $\langle q_{i-1}, \nu_{i-1} + t_i\rangle \xrightarrow{a_i} \langle q_i, \nu_i\rangle$ qui correspond au passage de l'état q_{i-1} à l'état q_i telle que $(\nu_{i-1} + t_i) \models \phi_i$ et $\nu_i = [X_i := 0](\nu_{i-1} + t_i)$.

On note $\omega(i) = q_i$ la position i de l'exécution ω. La durée d'une exécution notée $Dur(\omega)$ est définie par $\sum_{k=0}^{p-1} t_i$. L'exécution est étiquetée par le mot temporisé $w = (a_1, t_1)\cdots(a_p, t_p)$. Si le chemin P est acceptant ($q_p \in F$), on dit que le mot w est accepté par l'automate temporisé. L'ensemble de tous les mots temporisés acceptés par $\mathcal{A}$ est noté $L(\mathcal{A})$. On note $Path(\mathcal{A})$ l'ensemble des exécutions de $\mathcal{A}$.

Un état q est dit *accessible* s'il existe une valuation ν et une exécution finie de $\mathcal{A}$ de l'état initial $q_0 \in I$ jusqu'à $\langle q, \nu\rangle$. L'ensemble des états accessibles est noté $Reach(\mathcal{A})$.

1.1.2 Graphe des régions

Pour tout $t \in \mathbb{R}$, on note la partie fractionnaire de t par $fract(t)$ et $\lfloor t \rfloor$ la partie entière de t. Ainsi, $t = \lfloor t \rfloor + fract(t)$. On suppose que chaque horloge de $\mathcal{X}$ apparaît dans au moins une contrainte temporelle.

Définition 1.3. *Soit $\mathcal{A} = (\Sigma, Q, T, I, F, \mathcal{X}, inv)$ un automate temporisé. Pour tout $x \in \mathcal{X}$, soit c_x le plus grand entier c tel que les formules $(x \leq c)$ ou $(c \leq x)$ soient des sous formules dans les contraintes temporelles d'une transition $t \in T$.*
La relation [2] d'équivalence $\sim$ est définie sur l'ensemble des valuations temporelles pour $\mathcal{X}$. $\nu \sim \nu'$ ssi toutes les conditions suivantes sont vérifiées :

- *Pour tout $x \in \mathcal{X}$, soit $\lfloor \nu(x) \rfloor$ et $\lfloor \nu'(x) \rfloor$ sont égales, soit $\nu(x)$ et $\nu'(x)$ sont tous deux supérieurs à c_x.*
- *Pour tous $x, y \in \mathcal{X}$ telles que $\nu(x) \leq c_x$ et $\nu(y) \leq c_y$, $fract(\nu(x)) \leq fract(\nu(y))$ ssi $fract(\nu'(x)) \leq fract(\nu'(y))$.*
- *Pour tous $x \in \mathcal{X}$ telles que $\nu(x) \leq c_x$, $fract(\nu(x)) = 0$ ssi $fract(\nu'(x)) = 0$.*

Une *région* est une classe d'équivalence d'un ensemble de valuations temporelles, induite par $\sim$. On note $[\nu]$ la classe d'équivalence à laquelle la valuation ν appartient.
Pour une contrainte temporelle δ donnée de $\mathcal{A}$, si $\nu \sim \nu'$alors ν satisfait δ ssi ν' satisfait δ. On dit alors que la région α satisfait une contrainte temporelle δ ssi chaque $\nu \in \alpha$ satisfait δ. Chaque région peut être représentée en spécifiant :

1. pour tout $x \in \mathcal{X}$, une contrainte temporelle de l'ensemble

$$\{x = c \mid c = 0, 1, \cdots c_x\} \cup \{c - 1 < x < c \mid c = 1, \cdots c_x\} \cup \{x > c_x\},$$

2. pour toute paire d'horloge x et y tel que $c - 1 < x < c$ et $d - 1 < y < d$ apparaissent dans 1. pour c, d donnés, on a $fract(x) < fract(y)$, $fract(x) = fract(y)$ ou $fract(x) > fract(y)$.
   Enfin, le nombre de régions [2] sur $\mathbb{T}^{\mathcal{X}}$ est fini et borné par $[|\mathcal{X}|!.2^{|\mathcal{X}|}.\prod_{x\in\mathcal{X}}(2c_x+2)]$. Ce résultat correspond à l'ensemble des combinaisons possibles de formules dont la forme est décrite précédemment.

Définition 1.4. *Soit $c \in \mathbb{N}$. On dit que deux valuations ν et ν' sont c-équivalentes si :*
- *pour toute horloge x, on a $\nu(x) = \nu(x')$ ou bien $\nu(x) > c$ et $\nu(x') > c$*
- *pour toute paire d'horloges x, y, on a $\nu(x) - \nu(y) = \nu'(x) - \nu'(y)$ ou bien $\nu(x) - \nu(y) > c$ et $\nu'(x) - \nu'(y) > c$.*

A un automate temporisé, on associe un automate des régions $\mathcal{R}(\mathcal{A})$ où les états sont de la fome $\langle q, \alpha \rangle$ avec $q \in Q$ et α une région. L'automate des régions possède un état initial $\langle q_0, \alpha_0 \rangle$ et il existe une transition étiquetée par $a \in \Sigma$ de $\langle q, \alpha \rangle$ à $\langle q', \alpha' \rangle$ si et seulement si l'automate $\mathcal{A}$ est dans l'état q avec une certaine valuation $\nu \in \alpha$ et peut passer à l'état valué $\langle q', \nu' \rangle$ pour $\nu' \in \alpha'$, en laissant éventuellement s'écouler le temps en s puis en tirant la transition étiquetée par a.
L'automate des régions $\mathcal{R}(\mathcal{A})$ peut être vu comme un quotient de $\mathcal{A}$.

1.1.3 Graphe des zones

1.1.3.1 Définitions

Un $\mathcal{X} - hyperplan$ [34] est un ensemble de valuations ν qui satisfont une contrainte temporelle atomique dans $\mathcal{C}(\mathcal{X})$. La classe $\mathcal{H}_{\mathcal{X}}$ de polyèdres sur $\mathcal{X}$ est le plus petit ensemble de $2^{\mathbb{T}^{\mathcal{X}}}$ qui contient tous les $\mathcal{X} - hyperplans$ et est stable par union, intersection et complémentation [34]. Une conjonction de contraintes temporelles de $\mathcal{C}(\mathcal{X})$ forme donc un polyèdre sur $\mathcal{X}$. D'autre part, si Z n'est pas convexe [1], il peut alors être écrit sous forme d'union de k polyèdres convexes Z_i. On note $convexe(Z)$ l'ensemble $\{Z_1, \cdots, Z_k\}$.

Définition 1.5. (*Zone.*) *Une* zone *[10] est un sous ensemble de $\mathbb{T}^n$ défini par une conjonction de contraintes temporelles (polyèdre convexe sur $\mathcal{X}$). Soit $c \in \mathbb{N}$. Une zone c-bornée est une zone définie par des contraintes temporelles c-bornées. Soit Z une zone. L'ensemble des zones c-bornées contenant Z est fini et non vide ($\mathbb{T}^n$ est une zone c-bornée et contient Z), l'intersection de ces zones c-bornées est une zone c-bornée contenant Z et sera ainsi la plus petite zone ayant cette propriété. Cette zone est appelée la c-approximation de Z et est notée $close(Z, c)$.*

La c-approximation de Z correspond à Z', le plus petit polyèdre sur $\mathcal{X}$ contenant Z, tel que pour tout $\nu' \in Z'$ il existe $\nu \in Z$ telle que ν et ν' soient c-équivalentes. Z' est obtenu en ignorant toutes les contraintes temporelles qui impliquent des constantes supérieures à c (comme par exemple des contraintes de la forme $x \leq d$ ou $x < d$ avec $c \leq d$).
On définit par ailleurs les opérations $Z[Y := 0]$ et $[Y := 0]Z$ sur les zones qui correspondent à la remise à zéro *en avant* et *en arrière* des horloges appartenant à Y :

$$Z[Y := 0] = \{\nu[Y := 0] \mid \nu \in Z\}$$

$$[Y := 0]Z = \{\nu \mid \nu[Y := 0] \in Z\}$$

$Z[Y := 0]$ contient les valuations qui peuvent être obtenues à partir de certaines valuations de Z en remettant à zéro les horloges de Y. $[Y := 0]Z$ contient les valuations qui après avoir remis à zéro les horloges de Y appartiennent à Z.

[1] Un polyèdre Z est dit convexe si pour toute valuation ν_1,$\nu_2 \in Z$,pour tout $0 < \delta < 1$, on a $\delta\nu_1 + (1-\delta)\nu_2 \in Z$. Un polyèdre est convexe ssi il est défini par l'intersection finie d'un nombre fini d'hyperplans.

D'autre part, on définit de même les opérations *en avant* et *en arrière* de l'écoulement du temps. Pour un polyèdre sur $\mathcal{X}$, on définit ainsi les polyèdres $\swarrow Z$ et $\nearrow Z$ comme suit :

$$\nu \in \swarrow Z \text{ ssi } \exists t \in \mathbb{R} \text{ tel que } \nu + t \in Z$$

$$\nu \in \nearrow Z \text{ ssi } \exists t \in \mathbb{R} \text{ tel que } \nu - t \in Z$$

1.1.3.2 Description du graphe des zones associé à un automate temporisé

L'ensemble des polyèdres relatifs aux contraintes temporelles atomiques déduites d'un automate temporisé forme le graphe des zones associé à cet automate. La progression du temps permet de définir le comportement de cet automate par la construction du graphe des zones et l'utilisation des opérateurs $post()$ et $pre()$ que nous allons définir ci-dessous. Ce comportement sera décrit par son automate des zones, semblable de par la forme à l'automate des régions sauf que les régions y seront remplacées par des zones.
Ainsi, tout comme l'automate des régions $\mathcal{R}(\mathcal{A})$ associé à l'automate temporisé $\mathcal{A}$, nous définirons l'automate des zones $\mathcal{Z}(\mathcal{A})$ dont les états sont de la forme (q, Z) où q est un état de l'automate $\mathcal{A}$ et Z une zone (polyèdre convexe sur $\mathcal{X}$). Pour un état $S = (q, Z)$ de l'automate des zones dit état symbolique, deux transitions $e_1 = (q, Z_1, X_1, q_1)$ et $e_2 = (q_2, Z_2, X_2, q)$ de $\mathcal{A}$ et un entier naturel $c \geq c_{max}(\mathcal{A})$, on définit les opérateurs successeur et prédécesseur :

$$post(S, e_1) = (q_1, close(\nearrow ((Z \cap Z_1)[X_1 := 0]), c))$$

$$pre(S, e_2) = (q_2, ([X_2 := 0](\swarrow Z) \cap Z_2))$$

$post()$ contient tous les états (et leurs c-équivalents) qui peuvent être atteints d'un état $(q, \nu) \in (q, Z)$ en tirant la transition e et en laissant ensuite un certain temps s'écouler ; $pre()$ contient tous les états qui peuvent atteindre un état $(q, \nu) \in (q, Z)$ en tirant la transition e et en laissant ensuite un certain temps s'écouler. Si Z n'est pas convexe, on considère l'enveloppe convexe de Z, $convexe(Z)$. Dans la suite, la constante c sera la plus grande constante c_{max} apparaissant dans l'ensemble des contraintes temporelles d'un automate $\mathcal{A}$.

1.1.3.3 Algorithme d'accessibilité

L'un des algorithmes d'analyse en avant des automates temporisés fait appel à la c-approximation. q_0 est l'état initial de l'automate tandis que Z_0 représente la zone initiale qui est en général définie par la remise à zéro de toutes les horloges dans $\mathcal{X}$.

Cet algorithme termine car le nombre de zones c-bornées est fini et un nombre fini de c-approximations de zones peut ainsi être calculé pour tout état de $\mathcal{A}$. Cet algorithme calcule pas à pas un sur-ensemble de l'ensemble des états accessibles réels de $\mathcal{A}$ et teste ensuite si ce sur-ensemble intersecte ou non l'ensemble des états finaux de $\mathcal{A}$. Ainsi, si la réponse est "Non", aucun état final ne peut être atteint. Si la réponse est "Oui", le sur-ensemble pourrait en effet intersecter l'ensemble des états finaux sans que l'ensemble exact des états accessibles n'intersecte cet ensemble. Cet algorithme est *correct par rapport à l'accessibilité* si l'ensemble des états $q \in Q$ obtenus par l'algorithme est précisément égal à l'ensemble des états accessibles de l'automate $\mathcal{A}$.

Algorithme 1 *Accessibilité*($\mathcal{A}$)

```
    // Définir c comme étant égale à c_max de A
  Visit := ∅ ;
    // Correspond aux états symboliques visités
  Waiting := {(q_0, close(Z_0, c))} ;
  repeat
    Take and remove (q, Z) from Waiting
    if q final then
      return "YES" ;
    else
      if there is no (q, Z') ∈ Visit tel que Z ⊆ Z' then
        Visit := Visit ∪ {(q, Z)} ;
        Successor := {(q', close(post(Z, e), c)) tel que e ∈ out(q)} ;
        Waiting := Waiting ∪ Successor ;
      end if
    end if
  until Waiting = ∅ ;
  return "NO" ;
```

1.1.4 Discrétisation du temps

La discrétisation du temps [8, 9] est une autre représentation que l'on peut associer à un automate temporisé $\mathcal{A} = (\Sigma, Q, T, I, F, \mathcal{X}, inv)$. Les techniques utilisées pour la vérification de systèmes discrétisés sont décrites dans [19, 21].

En effet, cette méthode est différente de la construction des automates des régions ou des zones car le temps est représenté sous forme discrète et non pas dense. De plus, les gardes ϕ sont des gardes fermées, de la forme

$$\phi ::= x \sim c \mid \phi \wedge \phi \mid true$$

où $x, y \in \mathcal{X}$, $c \in \mathbb{Z}$ et $\sim \in \{\leq, \geq\}$.

Les valuations temporelles de l'automate associent à chaque horloge $x \in \mathcal{X}$ une valeur dans $\mathbb{N}$. L'écoulement du temps est représenté par des transitions de durée 1.

La discrétisation du temps permet finalement d'obtenir un processus de décision markovien (voir section 1.3) dont les transitions représentent soit l'écoulement du temps soit une transition $e \in T$ de l'automate $\mathcal{A}$.

1.2 Les Chaînes de Markov à temps discret

Une *distribution* (probabilité discrète) sur un ensemble fini Q est une fonction $\mu : Q \to [0, 1]$ tel que $\sum_{q \in Q} \mu(q) = 1$. Notons $support(\mu)$ le sous-ensemble de Q formé par les états q tels que $\mu(q) > 0$. Si Q' est un sous-ensemble de Q, alors $\mu(Q') = \sum_{q \in Q'} \mu(q)$. Pour tout $q \in Q$, la distribution ponctuelle μ_q désigne la distribution qui associe la probabilité 1 à l'élément q. Pour un ensemble dénombrable Q_∞, soit $Dist(Q_\infty)$ l'ensemble des distributions sur les sous-ensembles finis de Q_∞.

Définition 1.6. *(Chaîne de Markov.) Une* chaîne de Markov *homogène à temps discret (CMTD) [25, 23, 12] est un uplet* $(Q, \bar{q}, prob)$ *où l'on a :*
- Q *un ensemble dénombrable d'états dont l'état initial est* $\bar{q}$ *;*
- $prob : Q \to Dist(Q)$ *une fonction qui associe à chaque état* $q \in Q$ *une distribution dans* $Dist(Q)$ *;*

Remarque 1.2. *Une chaîne de Markov homogène à temps discret est définie par la matrice de transitions probabilistes* $P : Q \times Q \to [0, 1]$ *tel que* $\sum_{q' \in Q} P(q, q') = 1$ *pour tout état* $q \in Q$ *et* $P(q, q') = prob(q)(q')$.

1.3 Les processus de décision markoviens

Définition 1.7. *(Processus de décision markovien.) Un* processus de décision markovien *(PDM) [25, 30] est un uplet* $(Q, \bar{q}, prob)$ *où :*
- Q *est un ensemble fini d'états dont l'état initial est* $\bar{q}$ *;*
- $prob : Q \to \mathcal{P}_{fn}(Dist(Q))$ *est une fonction qui associe à chaque état* $q \in Q$ *un ensemble fini de distributions ;*

Remarque 1.3. *Un processus de décision markovien est un ensemble de chaînes de Markov avec un choix non déterministe d'une distribution.*

Le parcours de l'ensemble d'états du PDM est déterminé par le choix des transitions permises par la fonction *prob*. Ainsi, dans un état donné q, une transition est tirée en prenant tout d'abord de manière non déterministe une distribution p de l'ensemble $prob(q)$ et en faisant ensuite un choix probabiliste suivant p qui déterminera l'état cible q'. La transition sera notée $q \xrightarrow{p} q'$.
Dans certaines variantes des processus de décision markoviens, la définition de la fonction *prob* sera augmentée d'un ensemble d'événements (ou actions) Σ tel que $prob : Q \to \mathcal{P}_{fn}(\Sigma \times Dist(Q))$. La fonction *prob* associe ainsi une paire (σ, p) composée d'un événement σ et d'une distribution p à tout état $q \in Q$. La transition sera alors notée $q \xrightarrow{\sigma,p} q'$.

1.4 Les automates temporisés probabilistes paramétrés (ATPP)

1.4.1 Définition

Un Automate Temporisé Paramétré [3] est défini comme un automate temporisé où l'on peut avoir des paramètres (i.e des inconnues) à la place des constantes. En donnant à chaque paramètre une valeur concrète, on obtient un automate temporisé. Un automate temporisé paramétré est donc un ensemble d'automates temporisés.
Soit $\mathcal{P}$ un ensemble de paramètres. On note $\mathcal{C}_{\mathcal{P}}(\mathcal{X})$ l'ensemble des contraintes temporelles dont les contraintes atomiques sont de la forme $x \sim c$ où c représente soit un paramètre dans $\mathcal{P}$ soit un entier naturel et $\sim \in \{<, >, \leq, \geq\}$.

En ajoutant à ce type d'automates un facteur probabiliste au niveau des transitions, nous obtenons des Automates Temporisé Probabiblistes Paramétrés (ATPP) dont nous donnons la définition suivante :

Définition 1.8. *(Automate temporisé probabiliste paramétré.) Un* automate temporisé probabiliste paramétré *est un uplet*
$\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, inv, prob, \langle \tau_q \rangle_{q \in Q})$ *tels que :*

- *un ensemble fini Q d'états ;*
- *un ensemble fini $\mathcal{P}$ de paramètres ;*
- *un état initial $\bar{q} \in Q$;*
- *un ensemble fini $\mathcal{X}$ d'horloges ;*
- *une fonction $inv : Q \to \mathcal{C}_{\mathcal{P}}(\mathcal{X})$ qui associe à chaque état une contrainte temporelle appelée invariant ;*
- *une fonction $prob : Q \to \mathcal{P}_{fn}(Dist(2^{\mathcal{X}} \times Q))$ qui associe à chaque état un ensemble fini et non vide de distributions sur $2^{\mathcal{X}} \times Q$;*
- *une famille de fonctions $\langle \tau_q \rangle_{q \in Q}$, où pour tout état $q \in Q$, $\tau_q : prob(q) \to \mathcal{C}_{\mathcal{P}}(\mathcal{X})$ associe à tout $p \in prob(q)$ une condition d'activation qui est une contrainte temporelle appelée garde.*

Remarque 1.4. *Comme pour les automates temporisés de la définition 1.2, on note l'ensemble $I \subseteq Q$ le sous ensemble d'états initiaux d'un automate temporisé probabiliste paramétré et $F \subseteq Q$ son sous ensemble d'états finaux.*

Définition 1.9. *Soit $\mathcal{P}$ l'ensemble des paramètres de l'automate temporisé probabiliste paramétré $\mathcal{A}$. Une valuation paramétrique $\kappa : \mathcal{P} \to \mathbb{T}$ est une application qui associe à chaque paramètre $a \in \mathcal{P}$ une valeur dans $\mathbb{T}$ où $\mathbb{T}$ est égale à $\mathbb{Q}^+$ ou $\mathbb{R}^+$.*

Définition 1.10. *Pour un automate temporisé probabiliste paramétré $\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, inv, prob, \langle \tau_q \rangle_{q \in Q})$ muni d'une valuation paramétrique $\kappa \in \mathbb{T}^{\mathcal{P}}$, on définit un automate probabiliste temporisé $\mathcal{A}_\kappa = (Q, \bar{q}, \mathcal{X}, inv_\kappa, prob, \langle \tau_q^\kappa \rangle_{q \in Q})$ tel que pour toute transition $e = (q, g, p, X, q')$ de $\mathcal{A}$, on définit une transition e_κ de $\mathcal{A}_\kappa$ de la forme $e_\kappa = (q, \kappa(g), p, X, q')$ avec $\kappa(g) \in \mathcal{C}(\mathcal{X})$, $inv_\kappa(q) = \kappa(inv(q))$ et $inv_\kappa(q') = \kappa(inv(q'))$ avec $inv_\kappa(q)$, $inv_\kappa(q') \in \mathcal{C}(\mathcal{X})$.*

On note $\mathbb{T}^{\mathcal{P}}$ l'ensemble des valuations paramétriques κ de $\mathcal{P}$ dans $\mathbb{T}$.
Un automate temporisé probabiliste paramétré $\mathcal{A}$ sur l'ensemble de paramètres $\mathcal{P}$ représente l'ensemble d'automates temporisés probabilistes $\{\mathcal{A}_\kappa, \kappa \in \mathbb{T}^{\mathcal{P}}\}$.
Pour un automate temporisé probabiliste paramétré $\mathcal{A}$ muni d'une valuation paramétrique κ, le comportement de $\mathcal{A}$ est décrit par l'automate temporisé $\mathcal{A}_\kappa$.

1.4.2 Automates temporisés paramétrés

Les automates temporisés paramétrés peuvent aussi être vus comme des automates temporisés probabilistes paramétrés où le choix de l'état suivant à partir de l'état courant se fait de manière non déterministe et non plus probabiliste par le choix d'une distribution.

En reprenant ainsi la définition 1.8 des ATPPs, nous définissons les automates temporisés paramétrés de la manière suivante :

Définition 1.11. *(Automate temporisé paramétré.) Un automate temporisé probabiliste paramétré $\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, inv, T)$ est un* automate temporisé paramétré *[3] si $T \subseteq Q \times [\mathcal{C}_{\mathcal{P}}(\mathcal{X}) \times 2^{\mathcal{X}}] \times Q$.*

Définition 1.12. *(Valuation paramétrique cohérente.) On dit que la valuation paramétrique κ est cohérente avec l'automate temporisé paramétré $\mathcal{A}$ si pour un état initial $q_0 \in I$ et un état final $q_n \in F$ il existe une exécution*

$$\omega = \langle q_0, \nu_0 \rangle \xrightarrow[t_0]{g_1, X_1} \cdots \xrightarrow[t_{n-1}]{g_{n-1}, X_{n-1}} \langle q_n, \nu_n \rangle$$

dans l'automate $\mathcal{A}_\kappa$ avec $\omega \in Path(\mathcal{A}_\kappa)$.

On note $\Gamma(\mathcal{A})$ l'ensemble des valuations paramétriques cohérentes avec $\mathcal{A}$. L'ensemble $\Gamma_{q_n}(\mathcal{A}) = \{\kappa \mid q_n \in Reach(\mathcal{A}_\kappa)\}$ représente les valuations paramétriques telles que $q_n \in Reach(\mathcal{A})$ en partant d'un état initial $q_0 \in I$.
Etant donné un automate temporisé paramétré $\mathcal{A}$ et un état q_n, le problème d'accessibilité qui consiste à savoir si l'ensemble $\Gamma_{q_n}(\mathcal{A})$ est vide ou pas, est dans le cas général indécidable [3, 18]. Le problème est cependant décidable dans le cas où une unique horloge de $\mathcal{X}$ est comparée à des paramètres ce qui veut dire que dans toute contrainte temporelle de la forme $y \sim c$ où c est un paramètre, y représente toujours la même horloge dans $\mathcal{X}$ [3, 18].

Théorème 1.1. *Pour un automate temporisé paramétré $\mathcal{A}$, la question [3] de savoir si $\Gamma_{q_n}(\mathcal{A})$ est vide ou non est récursivement énumérable (i.e semi-décidable).*

Théorème 1.2. *Pour un automate temporisé paramétré $\mathcal{A}$ dont une seule horloge $x \in \mathcal{X}$ est comparée à des paramètres [3], si $\mathbb{T} = \mathbb{N}$ alors l'ensemble $\Gamma_{q_n}(\mathcal{A})$ est défini par une formule linéaire et tester si $\Gamma(\mathcal{A}) = \emptyset$ est décidable.*

Pour un automate temporisé paramétré $\mathcal{A}$, on cherche à déterminer le sous ensemble $\Delta \subseteq \mathbb{T}^{\mathcal{P}}$ de valuations paramétriques qui correspond à certaines spécifications auxquelles doit répondre le système modélisé par l'automate temporisé paramétré $\mathcal{A}$. Ces spécifications peuvent dans certains cas mener à restreindre l'ensemble des valeurs des paramètres. Toute valuation paramétrique n'appartenant pas à Δ ne répond pas aux spécifications en question. On crée un état puit, noté q_{sink}, que toute exécution répondant aux spécifications ne doit pas atteindre. On définit Δ comme l'ensemble des valuations κ telles que $\Delta \cap \Gamma_{q_{sink}}(\mathcal{A}) = \emptyset$.
L'outil HyTech [20] permet l'analyse de systèmes hybrides [20] y compris celle des automates temporisés paramétrés. Pour un automate temporisé paramétré $\mathcal{A}$, considérons un de ses paramètres $a \in \mathcal{P}$. Pour une valuation $\kappa : \mathcal{P} \to \mathbb{N}$, on attribue au paramètre a la valeur $\kappa(a)$ dans toute exécution $\omega \in Path(\mathcal{A})$. La valeur de a varie pour l'ensemble $Path(\mathcal{A})$ suivant la valuation paramétrique κ que l'on choisit.

Définition 1.13. *(Valeur sûre d'un paramètre.) Une valeur v est dite sûre pour un paramètre $a \in \mathcal{P}$ dans le cas où l'état q_{sink} n'est pas atteint lorsque la valeur v est attribuée à a.*

Une valeur v est donc sûre pour un paramètre a de l'automate $\mathcal{A}$ s'il n'existe aucune exécution $\omega \in Path(\mathcal{A})$ tel que

1. $last(\omega) = q_{sink}$ et
2. le paramètre a possède la valeur v.

Ceci revient à dire que v est sûre pour a s'il n'existe aucune exécution telle que $last(\omega) = q_{sink}$ et $a = v$ en q_{sink}.

1.4.3 Automates temporisés probabilistes

Un automate temporisé probabiliste peut découler d'un automate temporisé probabiliste paramétré en associant à chaque paramètre $a \in \mathcal{P}$ une valeur dans $\mathbb{T}$ suivant une valuation paramétrique κ choisie dans $\mathbb{T}^{\mathcal{P}}$.

Nous reprenons ainsi la définition des ATPPs pour donner celle des automates temporisés probabilistes.

Définition 1.14. *(Automate temporisé probabiliste.) Un automate temporisé probabiliste paramétré $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, inv, prob, \langle \tau_q \rangle_{q \in Q})$ est un* automate temporisé probabiliste *si toutes les contraintes temporelles appartiennent à $\mathcal{C}(\mathcal{X})$.*

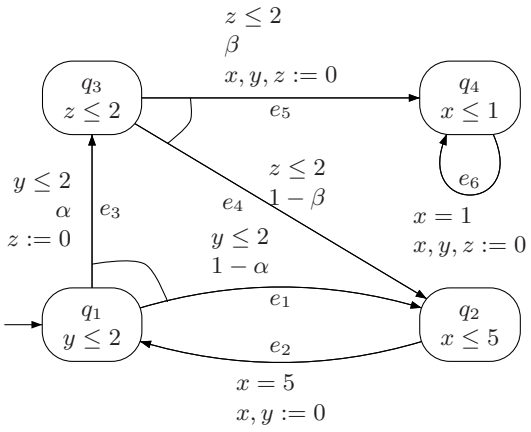

FIG. 1.1 – Automate temporisé probabiliste de l'exemple 1.1

Définition 1.15. *Une transition probabiliste d'un automate temporisé probabiliste est un uplet de la forme (q, g, p, X, q') tel que $p \in prob(q)$, $g = \tau_q(p)$ et $p(X, q') > 0$. L'ensemble des transitions généré par (q, g, p) est de la forme (q, g, p, X, q') avec $(X, q') \in 2^{\mathcal{X}} \times Q$.*

Définition 1.16. *Pour une exécution finie*

$$\omega = \langle q_0, \nu_0 \rangle \xrightarrow{g_0, p_0, X_0} \langle q_1, \nu_1 \rangle \xrightarrow{g_1, p_1, X_1} \cdots \xrightarrow{g_{n-1}, p_{n-1}, X_{n-1}} \langle q_n, \nu_n \rangle$$

d'un automate temporisé probabiliste $\mathcal{A}$, on définit $Pr(\omega)$ sa probabilité par $Pr(\omega) = \prod_{k=0}^{n-1} prob(q_k)(X_k, q_{k+1})$.

1.4.4 Exemple d'automate temporisé probabiliste

Exemple 1.1. *Considérons l'automate temporisé probabiliste décrit dans la figure 1.1. Cet automate $\mathcal{A}$ donne un modèle simplifié du protocole BRP (Bounded Retransmission Protocol) [15, 14]. $\mathcal{A}$ possède quatre états qui sont q_1, q_2, q_3 et q_4 et trois horloges x, y et z. Les probabilités sont représentées par les valeurs $\alpha = \frac{1}{3}$, $1 - \alpha = \frac{2}{3}$, $\beta = \frac{2}{3}$, $1 - \beta = \frac{1}{3}$ et 1 (les probabilités égales à 1 ne sont pas représentées dans la figure pour simplification). Prenons l'exemple de l'état q_1. L'invariant en q_1 ($inv(q_1)$), est égal à $y \leq 2$. La fonction prob associe à q_1 une unique distribution p sur $Dist(2^{\mathcal{X}} \times Q)$: deux transitions probabilistes sont engendrées par cette distribution qui sont $(q_1, g, \frac{1}{3}, z := 0, q_3)$ et $(q_1, g, \frac{2}{3}, \emptyset, q_2)$ avec $g = \tau_{q_1}(p) = y \leq 2$. La figure 1.2 décrit le graphe des zones associé à l'automate de la figure 1.1 en faisant abstraction des probabilités.*

1.5 Les systèmes probabilistes temporisés

Dans [24], on discrétise le temps (voir section 1.1.4) dans le but de calculer des temps moyens de convergence minimal ou maximal dans des automates temporisés probabilistes. En partant d'un automate temporisé probabiliste, on obtient un processus de décision markovien où l'écoulement du temps y est représenté par des transitions de durée 1 et de probabilité 1. Plus généralement, on parle de système probabiliste temporisé lorsque le temps est représenté dans un processus de décision markovien.

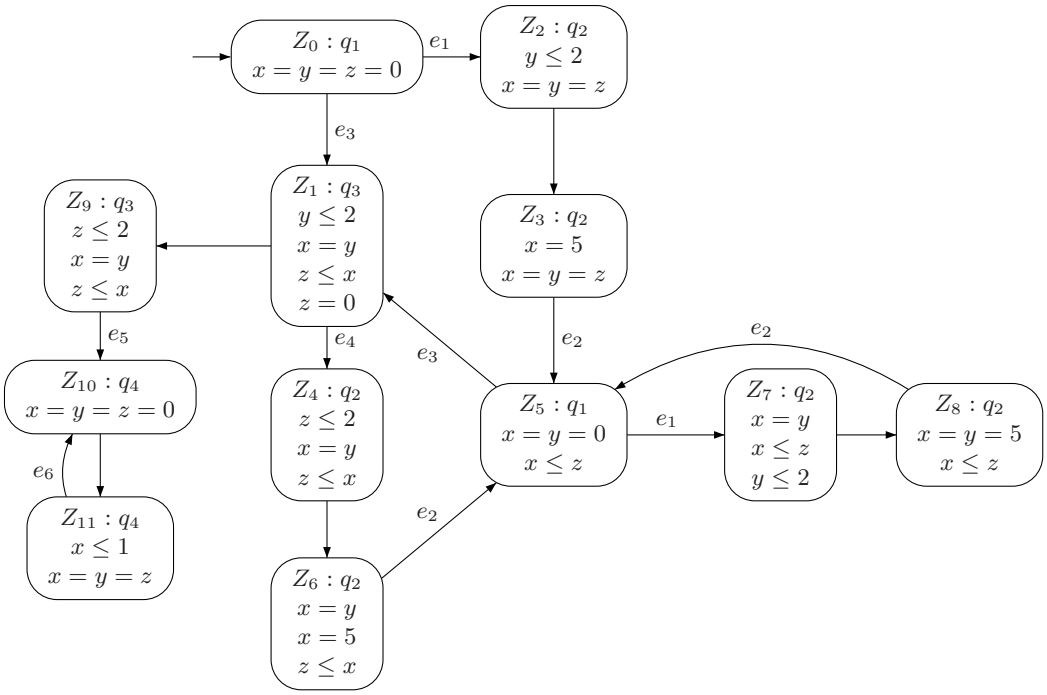

FIG. 1.2 – Graphe des zones de l'automate de l'exemple 1.1

Définition 1.17. *(Système probabiliste temporisé.) Un* système probabiliste temporisé *[25, 31]* $\mathcal{M}$ *est un processus de décision markovien* $(Q, \bar{q}, prob)$ *avec :*

- Q *est un ensemble dénombrable d'états dont l'état initial est* $\bar{q}$ *;*
- $prob : Q \to \mathcal{P}_{fn}(\mathbb{T} \times Dist(Q))$ *est une fonction qui attribue à chaque état* $q \in Q$ *un ensemble fini* $prob(q)$ *composé de paires de la forme* (t, p) *où* $t \in \mathbb{T}$ *et* $p \in Dist(Q)$*.*

$prob(q)$ représente l'ensemble des transitions qui peuvent être choisies de manière non déterministe dans l'état q. Chaque transition est de la forme (t, p), où t est la durée de la transition et p est la distribution sur l'ensemble des états successeurs. Ainsi, en faisant un choix non déterministe de (t, p) dans $prob(q)$ en q, une transition probabiliste est tirée après t unités de temps vers un état cible q' avec une probabilité égale à $p(q')$.
Les chemins dans un système probabiliste temporisé résultent d'un choix non déterministe et probabiliste en chaque état. Un *chemin* du système probabiliste temporisé $\mathcal{M} = (Q, \bar{q}, prob)$ est la séquence non vide finie ou infinie :

$$\omega = q_0 \xrightarrow{t_0,p_0} q_1 \xrightarrow{t_1,p_1} q_2 \xrightarrow{t_2,p_2} \cdots$$

où $q_i \in Q$, $(t_i, p_i) \in prob(q_i)$ et $p_i(q_{i+1}) > 0$ pour tout $0 \leq i \leq |\omega|$.
On note $Path_{fin}$ l'ensemble des chemins finis et $Path_{fin}(q)$ l'ensemble des chemins de $Path_{fin}$ tel que $\omega(0) = q$. $Path_{ful}$ est l'ensemble des chemins infinis et $Path_{ful}(q)$ l'ensemble des chemins de $Path_{ful}$ tel que $\omega(0) = q$.

Définition 1.18. *(Durée d'un chemin.) Pour tout chemin* ω *d'un système probabiliste [25, 31] temporisé* $\mathcal{M}$ *et tout* i *avec* $0 \leq i \leq |\omega|$*, on définit le* temps écoulé jusqu'à la ième transition

$\mathcal{D}_\omega(i)$ comme suit : $\mathcal{D}_\omega(0) = 0$ et pour tout $1 \leq i \leq |\omega|$ on a :

$$\mathcal{D}_\omega(i) = \sum_{j=0}^{i-1} t_j.$$

De plus, un chemin ω est dit divergent si pour tout $t \in \mathbb{R}$, il existe $j \in \mathbb{N}$ tel que $\mathcal{D}_\omega(j) > t$.

Définition 1.19. *Pour tout automate temporisé probabiliste $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, inv, prob, \langle \tau_q \rangle_{q \in Q})$, on définit le système probabiliste temporisé qui lui est associé $\mathcal{M}_\mathcal{A} = (Q_\mathcal{A}, \bar{q}, prob_\mathcal{A})$ [25] avec :*

- *Un état de $\mathcal{M}_\mathcal{A}$ est une paire $\langle q, \nu \rangle$, où $q \in Q$ est un état de l'automate temporisé probabiliste et $\nu \in \mathbb{T}^\mathcal{X}$ est une valuation telle que ν satisfait $inv(q)$. Soit $Q_\mathcal{A}$ l'ensemble dénombrable des états de $\mathcal{M}_\mathcal{A}$.*
- *La fonction $prob_\mathcal{A} : Q_\mathcal{A} \to \mathcal{P}_{fn}(\mathbb{T} \times Dist(Q_\mathcal{A}))$ associe à chaque état de $Q_\mathcal{A}$ un ensemble de transitions dont chacune est une paire $(t, \tilde{p})$ avec $t \in \mathbb{T}$, une durée et $\tilde{p} \in Dist(Q_\mathcal{A})$ une distribution sur l'ensemble des états $Q_\mathcal{A}$. Les transitions sont définies de deux manières. Pour tout $\langle q, \nu \rangle \in Q_\mathcal{A}$:*
  1. *Soit $(t, \tilde{p})$. $(t, \tilde{p}) \in prob_\mathcal{A}(\langle q, \nu \rangle)$ s'il existe $p \in prob(q)$ tel que :*
     (a) *la valuation $\nu + t$ satisfait la garde $\tau_q(p)$,*
     (b) *la valuation $\nu + t'$ satisfait l'invariant $inv(q)$ pour tout $0 \leq t' \leq t$ et*
     (c) *pour tout $\langle q', \nu' \rangle \in Q_\mathcal{A}$:*

$$\tilde{p}\langle q', \nu' \rangle = \sum_{\substack{X \subseteq \mathcal{X} \\ (\nu+t)[X:=0] = \nu'}} p(q', X)$$

  2. *Soit $(t, \tilde{p})$. $(t, \tilde{p}) \in prob_\mathcal{A}(\langle q, \nu \rangle)$ si :*
     (a) *la valuation $\nu + t'$ satisfait $inv(q)$ pour tout $0 \leq t' \leq t$ et*
     (b) *pour tout $\langle q', \nu' \rangle \in Q_\mathcal{A}$:*

$$\tilde{p}\langle q', \nu' \rangle = \begin{cases} 1 & si\ \langle q', \nu' \rangle = \langle q, \nu + t \rangle \\ 0 & sinon \end{cases}$$

1.5.1 Adversaire et adversaire divergent d'un système probabiliste temporisé

Soit ω un chemin. Si ω est fini, on note $last(\omega)$ le dernier état de ω et $step(\omega, i)$ la *ième* transition de ω [25]. On donne tout d'abord la définition d'*adversaire* pour ensuite nous focaliser uniquement sur une classe particulière d'*adversaires* dits *divergents*. En effet, l'étude de cas de systèmes temps réel se fait généralement en analysant uniquement les comportements vérifiant la propriété de divergence en temps. Ainsi, on ne tient pas compte des chemins non divergents lors de la vérification du système car ils sont la traduction de comportements irréalisables par l'arrêt de la progression du temps au delà d'une certaine limite.

Définition 1.20. *(Adversaire.) Un adversaire (ou scheduler) d'un système probabiliste temporisé $\mathcal{M} = (Q, \bar{q}, prob)$ est une fonction A qui associe à chaque chemin ω de $\mathcal{M}$ une paire (t, p) tel que $A(\omega) \in prob(last(\omega))$. On note Adv l'ensemble des adversaires de $\mathcal{M}$.*

Pour un adversaire A d'un système probabiliste temporisé $\mathcal{M} = (Q, \bar{q}, prob)$, soient $Path^A_{fin}$ l'ensemble des chemins finis tel que $step(\omega, i) = A(\omega^{(i)})$ pour tout $1 \leq i \leq |\omega|$ et $Path^A_{ful}$ le sous ensemble de $Path_{ful}$ tel que $step(\omega, i) = A(\omega^{(i)})$ pour tout $i \in \mathbb{N}$.

On associe à chaque adversaire A une chaîne de Markov à temps discret (politique markovienne [30]) qui pourrait être considérée comme un sous-ensemble de chemins de $\mathcal{M}$. Formellement, si A est un adversaire du système probabiliste temporisé $\mathcal{M}$, alors $MC^A = (Path^A_{fin}, \mathbf{P}^A)$ est une chaîne de Markov d'état initial $\bar{q}$ avec :

$$\mathbf{P^A}(\omega, \omega') = \begin{cases} \mathrm{p(q)} & \text{si A}(\omega) = (t, p) \text{ et } \omega' = \omega \xrightarrow{t,p} q \\ 0 & \text{sinon} \end{cases}$$

Pour tout système probabiliste temporisé dont A est un adversaire, soit $\mathcal{F}^A_{Path}$ [25] la plus petite σ-algèbre [2] sur $Path^A_{ful}$ qui contient les ensembles suivants :

$$\{\omega \mid \omega \in Path^A_{ful} \textit{ et } \omega' \textit{ est un préfixe de } \omega\}$$

pour $\omega' \in Path^A_{fin}$.

On définira dans ce qui suit une mesure [25] $Prob^A$ sur la σ-algèbre $\mathcal{F}^A_{Path}$ en commençant par définir une fonction sur l'ensemble des chemins finis $Path^A_{fin}$.

Définition 1.21. *Soit A un adversaire d'un système probabiliste temporisé $\mathcal{M}$. Soit $Prob^A_{fin} : Path^A_{fin} \to [0, 1]$ l'application définie sur la longueur des chemins appartenant à $Path^A_{fin}$. Si $|\omega| = 0$, alors $Prob^A_{fin} = 1$.*
Soit $\omega' \in Path^A_{fin}$ un chemin fini de A. Si $\omega' = \omega \xrightarrow{t,p} q$ pour $\omega \in Path^A_{fin}$ alors :

$$Prob^A_{fin}(\omega') = Prob^A_{fin}(\omega).\mathbf{P}^A(\omega, \omega').$$

Définition 1.22. *La mesure $Prob^A$ sur $\mathcal{F}^A_{Path}$ est l'unique mesure telle que :*

$$Prob^A\{\omega \mid \omega \ \in Path^A_{ful} \textit{ et } \omega' \textit{ est un préfixe de } \omega\} = Prob^A_{fin}(\omega')$$

Définition 1.23. *(Adversaire Divergent.) Un* adversaire *A d'un système probabiliste $\mathcal{M} = (Q, prob)$ est divergent ssi :*

$$Prob^A\{\omega \mid \omega \in Prob^A_{ful} \textit{ et } \omega \textit{ est divergent}\} = 1$$

Soit $\mathcal{A}_{div}$ l'ensemble des adversaires divergents.

On peut définir l'ensemble des adversaires $Adv^{\mathcal{A}}$ de $\mathcal{M}_{\mathcal{A}}$ en utilisant la définition 1.19. Notons $Adv^{\mathcal{A}}_{div}$ l'ensemble des adversaires divergents de $\mathcal{A}$. Dans [16], des algorithmes sont proposés pour vérifier la présence d'adversaires divergents.

[2] plus petit ensemble de parties de $Path^A_{ful}$ stable par complémentaire et par union dénombrable, contenant la partie vide et tous les ensembles $\{\omega \mid \omega \in Path^A_{ful} \textit{ et } \omega' \textit{ est un préfixe de } \omega\}$ pour $\omega' \in Path^A_{fin}$ [23].

1.6 Les automates à coûts positifs

1.6.1 Définition

On donne dans ce qui suit la définition d'un automate à coûts positifs G [30, 11].

Définition 1.24. *(Automate à coûts positifs.) Un* automate à coûts positifs G *est un uplet* $(Q, \bar{q}, prob, cost)$ *tel que :*

- Q *est un ensemble fini d'états dont l'état initial est* $\bar{q}$ *;*
- $prob : Q \to \mathcal{P}_{fn}(Dist(Q))$ *est une fonction qui associe à chaque état* $q \in Q$ *un ensemble fini de distributions ;*
- $cost : T \to \mathbb{N}$ *est une fonction qui attribue à chaque transition* $t \in T$ *de la forme* $q \xrightarrow{p} q'$ *un coût positif* $cost(t)$ *noté* $cost(q, q')$.

Un automate à coûts positifs est donc un processus de décision markovien muni d'une fonction, la fonction *cost* qui assigne à chaque transition un coût.

Dans la suite, les automates à coûts positifs que l'on considère sont des chaînes de Markov G munis d'une fonction, la fonction *cost* qui assigne à chaque transition un coût.

Soit G un automate à coûts. On suppose que G possède un unique état final et absorbant f (voir la section 2.2).

On souhaite calculer le coût moyen de convergence vers un état final et absorbant f de cet automate en partant d'un état quelconque.

Tout chemin $\omega \in Path^G_{fin}(q_0)$ de la forme $q_0 \overset{p_0,c_0}{\to} q_1 \cdots \overset{p_{n-1},c_{n-1}}{\to} q_n$ possède ainsi un coût $cost(\omega) = \sum_{k=0}^{n-1} c_k$ avec $c_k = cost(q_k, q_{k+1})$ et une probabilité $Prob^G(\omega) = \prod_{k=0}^{n-1} p_k$ avec $p_k = P(q_k, q_{k+1})$ où P est la matrice de transitions probabilistes associée à G et définie dans la remarque . Dans le calcul qui suit, tout chemin $\omega \in Path^G_{fin}(q_0)$ est acceptant avec $q_n = f$ (voir la section 1.1).

Vu que f est l'unique état final et absorbant, on a $P(f, q_i) = 0$ pour tout $q_i \in Q$ avec $q_i \neq f$ et $P(f, f) = 1$.

1.6.2 Calcul du coût moyen de convergence : formule de Bertsekas

Les calculs de coût moyen de convergence dans les processus de décision markoviens sont proposés dans [6, 7]. On se restreint dans ce qui suit au calcul du coût moyen de convergence dans les chaînes de Markov.

Soit q_{i_0} un état d'une chaîne de Markov G. On présente le coût moyen de convergence en partant de q_{i_0} pour atteindre f dans G. Il est noté $cost_{moy}(q_{i_0})$.

$$
\begin{aligned}
& cost_{moy}(q_{i_0}) \\
= & \sum_{\omega \in Path^{\mathcal{M}}_{fin}(q_{i_0})} cost(\omega) Prob^G(\omega) \\
= & \sum_{\substack{q_{i0} \to q_{i_1} \\ q_{i_1} \neq f}} cost(q_{i_0}, q_{i_1}) P(q_{i_0}, q_{i_1}) \cdot \sum_{Path^{\mathcal{M}}_{fin}(q_{i_1})} P(q_{i_1}, q_{i_2}) \cdots P(q_{i_{n-1}}, f) \\
+ & \sum_{q_{i_0} \to f} cost(q_{i_0}, f) P(q_{i_0}, f) \\
+ & \sum_{\substack{q_{i_0} \to q_{i_1} \\ q_{i_1} \neq f}} P(q_{i_0}, q_{i_1}) \cdot \sum_{Path^{G}_{fin}(q_{i_1})} P(q_{i_1}, q_{i_2}) \cdots P(q_{i_{n-1}}, f) [cost(q_{i_1}, q_{i_2}) + \cdots + cost(q_{i_{n-1}}, f)] \\
= & \sum_{\substack{q_{i_0} \to q_{i_1} \\ q_{i_1} \neq f}} cost(q_{i_0}, q_{i_1}) P(q_{i_0}, q_{i_1}) \cdot \sum_{Path^{G}_{fin}(q_{i_1})} P(q_{i_1}, q_{i_2}) \cdots P(q_{i_{n-1}}, f) \\
+ & \sum_{q_{i_0} \to f} cost(q_{i_0}, f) P(q_{i_0}, f) \\
+ & \sum_{\substack{q_{i_0} \to q_{i_1} \\ q_{i_1} \neq f}} P(q_{i_0}, q_{i_1}) \cdot cost_{moy}(q_{i_1}).
\end{aligned}
$$

Or dans le cas d'une chaîne de Markov, on a

$$\sum_{Path^{G}_{fin}(q_{i_1})} P(q_{i_1}, q_{i_2}) \cdots P(q_{i_{n-1}}, f) = 1$$

D'où

$$
\begin{aligned}
& cost_{moy}(q_{i_0}) \\
= & \sum_{\substack{q_{i_0} \to q_{i_1} \\ q_{i_1} \neq f}} cost(q_{i_0}, q_{i_1}) P(q_{i_0}, q_{i_1}) \\
+ & \sum_{q_{i_0} \to f} cost(q_{i_0}, f) P(q_{i_0}, f) \\
+ & \sum_{\substack{q_{i_0} \to q_{i_1} \\ q_{i_1} \neq f}} P(q_{i_0}, q_{i_1}) \cdot cost_{moy}(q_{i_1}).
\end{aligned}
$$

On en tire la relation matricielle suivante :

$$T = PT + C$$

avec

- $T = (cost_{moy(}q_i))_{q_i \neq f}$ le vecteur coût dont chaque composante i représente le coût moyen de convergence de l'état q_i vers l'état final f pour tout $i \in \{0, \cdots, n-1\}$.
- $P = (P(q_i, q_j))_{q_i, q_j \neq f}$ la matrice de transition privée de l'état final.
- $C = Q + R$ tel que

$$Q_{q_i} = \sum_{\substack{q_i \to q_j \\ q_j \neq f}} cost(q_i, q_j) P(q_i, q_j)$$

et

$$R_{q_i} = \sum_{q_i \to f} cost(q_i, f) P(q_i, f).$$

Chapitre 2

Les automates temporisés probabilistes paramétrés et déterminés (ATPPD)

2.1 La progression du temps

La progression du temps dans un automate temporisé [34] peut être entravée par des comportements de blocage ou par des comportements zenon. Ces comportements ne sont pas désirables dans la description du fonctionnement normal de systèmes temps réel. Les comportements zenon sont en particulier en contradiction avec la notion de divergence temporelle souhaitée dans toute description de systèmes temps réel. En effet, la notion de blocage dans un automate temporisé $\mathcal{A}$ est traduite par l'impossibilité de tirer une transition t dont l'état source est l'état accessible q suivant une valuation ν donnée des horloges. Le blocage empêche ainsi la progression discrète de $\mathcal{A}$ en terme de transitions. D'autre part, le comportement zenon est un autre type de blocage qui se traduit par l'arrêt de la progression du temps.

2.1.1 Les automates temporisés probabilistes bien formés

2.1.1.1 Définition du blocage dans un automate temporisé

Définition 2.1. *(Etat bloquant par rapport à une valuation.) On dit qu'un état q est* bloquant par rapport à une valuation ν *(ou $\langle q, \nu\rangle$ est dit bloquant) s'il n'existe pas $t \in \mathbb{R}^+$ tel que $\langle q, \nu\rangle \xrightarrow[t]{g,a,X} \langle q', \nu'\rangle$ pour au moins une transition $e \in out(q)$ avec $e = (q, g, a, X, q')$ et $\nu' = (\nu + t)[X := 0]$.*

Définition 2.2. *(Etat bloquant.) Un état q dans un automate temporisé $\mathcal{A}$ est dit* bloquant *s'il existe une valuation ν tel que $\langle q, \nu\rangle$ est bloquant.*

Pour un état q de $\mathcal{A}$, on définit $free(q)$ comme l'ensemble des valuations ν tel que $\langle q, \nu\rangle$ n'est pas un état bloquant. On a

$$free(q) = \bigcup_{e \in out(q)} \swarrow (garde(e) \cap ([reset(e) := 0] invariant(cible(e)))).$$

Définition 2.3. *(Automate non bloquant.) Un* automate $\mathcal{A}$ *est* non bloquant *si pour tout* $q \in Reach(\mathcal{A})$*, s'il existe une valuation* ν *tel que* $\langle q, \nu \rangle$ *est accessible alors* $\langle q, \nu \rangle$ *est non bloquant et* $\nu \in free(q)$*.*

On peut ainsi donner une condition suffisante pour vérifier que l'automate $\mathcal{A}$ est non bloquant.

Lemme 2.1. *Soit* $\mathcal{A}$ *un automate. Si pour tout état* q *et pour tout* $e \in in(q)$*, la condition suivante est vérifiée :*

$$(\nearrow ((garde(e))[reset(e) := 0])) \cap invariant(q) \subseteq free(q)$$

Alors $\mathcal{A}$ *est non bloquant.*

On peut ainsi transformer tout automate $\mathcal{A}$ ayant des états accessibles (i.e dans $Reach(\mathcal{A})$) bloquants, en un autre automate $\mathcal{A}'$ non bloquant en imposant la condition locale $free(q)$ à chaque état $q \in Reach(\mathcal{A})$.

2.1.1.2 Algorithme de détection des états bloquants

Il existe un algorithme [34] qui permet de détecter les états bloquants d'un automate temporisé $\mathcal{A}$. Cet algorithme se base sur un algorithme d'accessibilité à la volée sur le graphe des zones correspondant à l'automate $\mathcal{A}$.
L' algorithme prend en entrée un état symbolique $S_0 = (q_0, Z_0)$, où q_0 est l'état initial de l'automate $\mathcal{A}$ et Z_0 une zone, et un ensemble cible d'états symboliques $\mathcal{Z}$. Il retourne une zone Z[1] avec $Z \subseteq Z_0$ telle que pour toute valuation $\nu \in Z$, on peut atteindre un état de l'ensemble $\mathcal{Z}$ en partant de l'état q_0 muni de la valuation ν. Si $Z = \emptyset$ alors aucune valuation de Z_0 ne nous permet d'atteindre un état de $\mathcal{Z}$ à partir de l'état initial q_0. L'ensemble $Visit$ est l'ensemble courant des états visités et est initialement vide.
Le parcours de l'automate des zones se fait à la volée grâce à l'opérateur *post*. En effet, cet opérateur permet de parcourir en avant l'automate des zones en calculant les successeurs d'un état symbolique donné. La fonction *Reach* est alors appelée récursivement sur chaque nouvel état symbolique atteint tant qu'aucun de ces états n'appartient à $\mathcal{Z}$. L'un des deux résultats suivant est renvoyé par *Reach* :

- Si, après exploration de l'automate des zones, aucun état atteint Z n'appartient à $\mathcal{Z}$, la fonction retourne $\emptyset$. Aucune valuation de Z_0 ne permet ainsi d'atteindre l'ensemble $\mathcal{Z}$.
- Dès qu'un état $S_l = (q_l, Z_l)$ tel qu'il existe $(q, Z) \in \mathcal{Z}$ et $q_l = q$ et $Z \cap Z_l \neq \emptyset$ est atteint, l'algorithme s'arrête. Il calcule un chemin parcouru π dans l'automate des zones de S_0 à S_l avec $\pi = S_0 \xrightarrow{e_0} \cdots \xrightarrow{e_{l-1}} S_l$ tel que S_i est de la forme (q_i, Z_i). L'ensemble ${Z_{i-1}}' = Z_{i-1} \cap pre(Z_i, q_i)$ calculé pour chaque état exploré du graphe des zones permettra de renvoyer le chemin parcouru entre q_0 et q_l dans l'automate $\mathcal{A}$. On calcule ainsi $Z \subseteq Z_0$ ($Z = Z'_0$) tel que $\forall \nu = \nu_0 \in Z$, $\exists \rho = \langle q_0, \nu_0 \rangle \xrightarrow{e_0} \langle q_1, \nu_1 \rangle \xrightarrow{e_1} \cdots \xrightarrow{e_{l-1}} \langle q_l, \nu_l \rangle$ tel que $\forall \nu_i \in {Z_i}'$, $\exists t_i$, tel que $\langle q_i, \nu_i \rangle \xrightarrow{e_i} \langle q_{i+1}, \nu_{i+1} \rangle$ avec $\nu_{i+1} = (\nu_i + t_i)[X := 0] \in Z'_{i+1}$.

Par ailleurs, il peut y avoir des états dans $S_0 \setminus S$ qui atteignent $\mathcal{Z}$ par un chemin différent ou identique à celui calculé pour S[2].

[1] Pour un état de l'automate des zones $S = (q, Z)$, nous utiliserons S dans certains cas pour désigner la zone associée Z.

[2] Pour plus de précision, nous dirons qu'il pourrait exister des valuations appartenant à $Z_0 \setminus Z$ qui permettraient d'atteindre l'ensemble cible $\mathcal{Z}$ en partant de l'état initial q_0.

Algorithme 2 $Reach(S_0, \mathcal{Z})$

```
  let S_0 = (q_0, Z_0);
if (∃(q, Z) ∈ Z | (q = q_0) ∧ (Z ∩ Z_0 ≠ ∅)) then
    return (q, Z ∩ Z_0) ;
end if
Visit := Visit ∪ {S_0} ;
for all (e ∈ out(q_0)) do
    (q_1, Z_1) := post(S_0, e) ;
    if (Z_1 = ∅) then
      continue ;
      //L'instruction continue permet de revenir directement dans la boucle for.
    else if (∃(q_1, Z_1') ∈ Visit | Z_1 ⊆ Z_1') then
      continue ;
    else
      S_1 := Reach((q_1, Z_1), Z) ;
      if (S_1 ≠ ∅) then
        return S_0 ∩ pre(S_1, e) ;
      end if
    end if
end for
return ∅;
}
```

Nous pouvons détecter les états bloquants en utilisant l'algorithme précédent. Etant donné un état q et une valuation ν, notons $S = (q, Z)$, tel que $\nu \in Z$, et $\mathcal{Z}$ l'ensemble des états symboliques $S = (q', Z')$ pour tout état cible q' des transitions $e = (q, g, a, X, q') \in out(q)$. L'algorithme $Reach(S, \mathcal{Z})$ permet alors de savoir si l'état $\langle q, \nu \rangle$ est bloquant ou non : il renvoie l'ensemble vide dans le cas où $\langle q, \nu \rangle$ est bloquant. Si l'état $\langle q, \nu \rangle$ est bloquant, on applique l'algorithme $Reach(S_0, (q, Z))$ pour connaître comment cet état est atteint.
De même, si $\mathcal{A} = (\Sigma, Q, T, I, F, \mathcal{X}, inv)$ est un automate, définissons

$$\mathcal{Z}^{\dagger} = \bigcup_{q \in Q} convexe(q, invariant(q) \setminus free(q))$$

l'ensemble d'états cibles. Alors l'algorithme $Reach((q_0, 0), \mathcal{Z}^{\dagger})$ permet de savoir si l'automate $\mathcal{A}$ est bloquant ou non, d'après le lemme 2.1. Ainsi, $\mathcal{A}$ est sans état bloquant ssi $Reach((q_0, 0), \mathcal{Z}^{\dagger})$ renvoie $\emptyset$. Par définition, tout état symbolique (q, Z) est tel que $Z \subseteq invariant(q)$, ainsi vérifier $Z \cap (invariant(q) \setminus free(q)) = \emptyset$ revient à vérifier que $Z \setminus free(q) = \emptyset$.

2.1.1.3 Automate temporisé probabiliste bien formé

Définition 2.4. *Un automate temporisé probabiliste $\mathcal{A}$ est dit bien formé si pour tout état q et toute valuation $\nu \in \mathbb{R}^{\mathcal{X}}$ tel que $\nu \models inv(q)$, il existe $t \in \mathbb{R}^{+}$ tel qu'au moins une transition probabiliste peut être tirée de q après que t unités de temps ne se soient écoulées en q. Formellement, un automate temporisé probabiliste $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, inv, prob, \langle \tau_q \rangle_{q \in Q})$ est*

bien formé si : $\forall(q,g,p) \in prob, \forall \nu \in \mathbb{R}^{\mathcal{X}}$ *telle que* $\nu \models g$ *alors*

$$(\forall(X,q') \in support(p)),\ \nu[X:=0] \models inv(q').$$

Un automate temporisé probabiliste quelconque peut être transformé en un autre automate temporisé probabiliste bien formé en remplaçant la garde g dans chaque transition $(q,g,p) \in prob$ par :

$$\bigwedge_{(X,q') \in support(p)} ([X:=0]inv(q')) \wedge g.$$

Il permet ainsi d'éviter qu'un état accessible ne soit bloquant. Cette restriction sur les gardes provient de la redéfinition de l'ensemble $free(q)$ (définition 2.2) dans la cas d'un automate temporisé probabiliste. On aura alors :

$$free(q) = \bigcup_{(q,g,p) \in prob} \nwarrow \left(\bigcap_{(X,q') \in support(p)} (g \cap ([X:=0]inv(q'))) \right).$$

L'ensemble $free(q)$ contient alors toutes les valuations ν telles que $\langle q, \nu \rangle$ n'est pas un état bloquant. Ces restrictions imposées aux gardes afin d'obtenir un automate bien formé suppriment éventuellement certains comportements de l'automate initial mais qui sont indésirables.

Exemple 2.1. *Reprenons l'automate de l'exemple 1.1 décrit par la figure 1.1. Cet automate temporisé probabiliste* $\mathcal{A}$ *donne un modèle simplifié du protocole BRP (Bounded Retransmission Protocol) [15, 14].* $\mathcal{A}$ *est bien formé car pour chaque état* q_i *tel qu'il existe une transition probabiliste* $(q_i, g, prob(q_i)(q_k), X, q_k)$*, nous pouvons montrer localement d'après le lemme 2.1 que*

$$(\nwarrow((g)[X:=0]))) \cap (inv(q_k)) \subseteq free(q_k).$$

Soit ω *une exécution de* $\mathcal{A}$ *:*

$$\omega = \langle q_1, \nu_1 \rangle \xrightarrow{t_1} \langle q_3, \nu_2 \rangle \xrightarrow{t_2} \langle q_2, \nu_3 \rangle \xrightarrow{t_3} \langle q_1, \nu_4 \rangle.$$

Prenons l'exemple de l'état q_2 *et montrons qu'il n'est pas bloquant. Supposons tout d'abord que* $\nu_2 \in free(q_3)$*. Il existe ainsi* $t_2 \in \mathbb{R}^+$ *telle que* $\langle q_3, \nu_2 \rangle \xrightarrow{t_2} \langle q_2, \nu_3 \rangle$ *avec* $\nu_3 = \nu_2 + t_2$*. La transition probabiliste* e_4 *peut ainsi être tirée et la valuation* ν_3 *vérifie l'invariant* $x \leq 5$*. Montrons que* $\nu_3 \in free(q_2)$*. Pour cela, il faut trouver* $t_3 \in \mathbb{R}^+$ *tel que* $\langle q_2, \nu_3 \rangle \xrightarrow{t_3} \langle q_1, \nu_4 \rangle$ *avec* $\nu_4 = (\nu_3 + t_3)[x, y := 0]$*. En effet, les horloges* x *et* y *sont remises à zéro en* e_2*. A l'instant où l'on entre dans l'état* q_3*, on a* $\nu_2(z) = 0$*,* $\nu_2(x) \leq 2$ *et* $\nu_2(y) \leq 2$*. Le temps maximal qui pourrait s'écouler en* q_3 *est donc égal à 2 unités de temps. Par ailleurs, pour tout* $t_2 \in [0,2]$*, la transition probabiliste* e_4 *est tirable. Ainsi,* $\nu_3(x)$ *est égal à* $(\nu_2 + t_2)(x)$ *qui est inférieur ou égal à 4 unités de temps lorsqu'on entre dans l'état* q_2*. Il existe donc* $t_3 \geq 0$ *tel que* $(\nu_3 + t_3)(x) = 5$ *et* $\nu_4(y) = (\nu_3 + t_3)[y := 0] = 0$ *vérifie l'invariant de l'état* q_1*. Notons que si la garde en* q_2 *est* $x < 4$*,* $\langle q_3, \nu_2 \rangle$ *pourrait être bloquant pour certaines valeurs de* t_2*. Il faudrait donc avoir une garde en* q_2 *de la forme* $x \leq T1$ *avec* $T1 > 4$*.*

Nous avons donc montré que pour toute valuation ν_3 *appartenant à* $(\nwarrow\ ((0 \leq z \leq 2)))$ *telle que* $\nu_3 \models (x \leq 2)$*,* $\nu_3 \in free(q_2)$*.*

2.1.2 Le comportement fortement non-zenon des automates temporisés

2.1.2.1 Définitions de convergence dans un automate temporisé

Définition 2.5. *(Exécution zenon.) Soit une exécution infinie ω :*

$$\omega = \langle q_0, \nu_0 \rangle \xrightarrow{t_0} \langle q_1, \nu_1 \rangle \cdots \xrightarrow{t_{p-1}} \langle q_{p-1}, \nu_{p-1} \rangle \cdots$$

Si $Dur(\omega) \neq \infty$ (i.e il existe $t \in \mathbb{R}$ tel que pour tout i, $\sum_{k=0}^{i-1} t_k < t$) alors ω est dite zenon.

Un état q d'un automate $\mathcal{A}$ est dit à *blocage temporel* (*"timelock"*) si toutes les exécutions infinies partant de q sont zenon. Un automate $\mathcal{A}$ est *sans blocage temporel* (ou *timelock-free*) si aucun de ses états accessibles n'est à blocage temporel. Les états dits "timelocks" empêchent l'écoulement réaliste du temps.

Lemme 2.2. *Un automate $\mathcal{A}$ est* sans blocage temporel *(ou* "timelock-free"*) ssi pour tout état $q \in Reach(\mathcal{A})$, il existe une exécution ω :*

$$\omega = \langle q, \nu \rangle \xrightarrow{t_0} \langle q_1, \nu_1 \rangle \cdots \xrightarrow{t_{i-1}} \langle q_i, \nu_i \rangle \cdots ,$$

partant de $\langle q, \nu \rangle$ avec $\langle q, \nu \rangle$ accessible et une position i de ω tel que $\sum_{k=0}^{i-1} t_k \geq 1$.

Définition 2.6. *(Automate fortement non zenon.) Un automate $\mathcal{A}$ est dit* fortement non zenon *si pour tout cycle $q_0 \xrightarrow{e_0} q_1 \xrightarrow{e_1} \cdots \xrightarrow{e_m} q_0$ il existe une horloge x et $0 \leq i, j \leq m$ tel que :*

- *x est remise à zéro à la transition i i.e $x \in reset(e_i)$ et*
- *x est minorée par 1 à la transition j avec $(x < 1) \cap garde(e_j) = \emptyset$.*

Ceci signifie qu'une unité de temps s'est au moins écoulée dans chaque cycle d'un automate $\mathcal{A}$ fortement non zenon.

Remarque 2.1. *Si un automate $\mathcal{A}$ est fortement non zenon alors il est non zenon. En effet, la propriété de fortement non zenon induit celle de non zenon car elle impose l'écoulement d'au moins une unité de temps dans chaque cycle, ce qui empêche le temps de converger sur une exécution infinie.*

2.1.2.2 Détection des états à blocage temporel (timelocks)

L'algorithme *Reach* ne renvoie pas tous les états S de S_0 desquels l'ensemble $\mathcal{Z}$ est accessible. En ce sens, S n'est donc pas maximal dans S_0. On peut compléter l'algorithme *Reach* [34] pour obtenir le plus grand sous-ensemble S de S_0 tel que aucun état de S ne peut atteindre $\mathcal{Z}$. Comme S n'est pas nécessairement convexe, il sera représenté par un ensemble d'états symboliques $\mathcal{Z}_0$. On définit ainsi une autre procédure *CompleteReach* [34] comme suit :

Algorithme 3 $CompleteReach(S_0, \mathcal{Z})$

$\mathcal{Z}_0 := \{S_0\}$;
while $(\exists S = (q, Z) \in \mathcal{Z}_0)$ **do**
  $S' := Reach(S, \mathcal{Z})$;
  $\mathcal{Z}_0 := (\mathcal{Z}_0 \setminus \{S\}) \cup convexe(S \setminus S')$;
end while
return $\mathcal{Z}_0$;

L'algorithme calcule l'ensemble des états $S' \in S$ desquels $\mathcal{Z}$ est atteint par la procédure *Reach*. Certains autres états dans $S \setminus S'$ pourraient aussi atteindre $\mathcal{Z}$, la procédure *Reach* est pour cela réitérée sur l'ensemble $convexe(S \setminus S')$.
Pour détecter les états à blocage temporel (timelock) dans un automate $\mathcal{A}$, on utilise les procédures *Reach* et *CompleteReach*.
Reach permet de générer l'ensemble des états symboliques accessibles dans $\mathcal{A}$. Pour tout état symbolique S, *CompleteReach* est appliquée sur une extension de l'automate $\mathcal{A}$ noté $\mathcal{A}^+$. $\mathcal{A}^+$ fait appel à une horloge auxilaire z pour trouver les états de S qui ne laissent pas le temps s'écouler pour une unité de temps. Pour un automate $\mathcal{A} = (\Sigma, Q, T, I, F, \mathcal{X}, inv)$, on définira donc $\mathcal{A}^+$ par $(\Sigma, Q, T, I, F, \mathcal{X} \cup \{z\}, inv)$.
Soit $\mathcal{Z}_{\geq 1}$ l'ensemble des états symboliques $\{(q, z \geq 1) \mid q \in S\}$.
L'algorithme de détection des états à blocage temporel (timelocks) est le suivant :

Algorithme 4 $TimelockReach(S)$

```
let S = (q, Z) ;
Z := CompleteReach_{A+}((q, Z ∩ (z = 0)), Z_{≥1}) ;
if (Z ≠ ∅) then
  return "YES" ;
end if
Visit := Visit ∪ {S} ;
for all (e ∈ out(q)) do
  (q1, Z1) := post(S, e) ;
  if (Z1 = ∅) then
    continue ;
  else if (∃(q1, Z1') ∈ Visit.Z1 ⊆ Z1') then
    continue ;
  else
    if (TimelockReach((q1, Z1) =" YES") then
      return "YES" ;
    end if
  end if
end for
return "NO" ;
```

La procédure $TimelockReach$ [34] permet de vérifier s'il existe des états dans S qui ne laissent pas passer une unité de temps le long d'une exécution. Pour ce test, on fait appel à la procédure *CompleteReach* sur $\mathcal{A}^+$, l'état initial $(q, Z \cap (z = 0))$ et l'ensemble des états symboliques cibles $\mathcal{Z}_{\geq 1}$. Ainsi, *CompleteReach* permet de calculer un ensemble d'états symboliques $\mathcal{Z}$ qui ne peuvent atteindre aucun état de $\mathcal{Z}_{\geq 1}$. L'appel $CompleteReach_{\mathcal{A}^+}((q, Z \cap (z = 0)), \mathcal{Z}_{\geq 1})$ permet donc de savoir si l'on peut atteindre un nouvel état symbolique cible de la forme $(q', Z' \cap (z \geq 1))$ à partir d'un état symbolique $(q, Z \cap (z = 0))$, ce qui permet de conclure qu'une unité de temps s'est au moins écoulée pour passer de $\langle q, \nu \rangle$ à $\langle q', \nu' \rangle$ où $\nu \in Z$ et $\nu' \in Z'$.
Si $\mathcal{Z}$ est non vide, l'ensemble S contient alors des états à blocage temporel (timelock). D'après le lemme 2.2, $\mathcal{A}$ est sans état à blocage temporel ssi $TimelockReach((q_0, 0))$ renvoie "NO".

Exemple 2.2. *Considérons l'automate décrit dans la figure 1.1 et son graphe des zones décrit dans la figure 1.2. Cet automate est fortement non zenon. Prenons comme exemple le cycle composé successivement des transitions probabilistes e_2, e_3 et e_4. L'horloge x est remise à zéro en e_2. Cet automate est fortement non zenon car $x \geq 1$ en e_2. Ce raisonnement s'applique aussi au cycle formé par les transitions probabilistes e_1 et e_2 car l'horloge x est remise à zéro en e_2 et $x \geq 1$ en e_2.*
Par ailleurs, pour un cycle

$$\omega = \langle q_0, \nu_0 \rangle \overset{e_0}{\rightarrow} \cdots \overset{e_{i-1}}{\rightarrow} \langle q_i, \nu_i \rangle \overset{e_i}{\rightarrow} \cdots \overset{e_{n-1}}{\rightarrow} \langle q_n, \nu_n \rangle$$

tel que e_{i-1} remet une horloge x à zéro et $q_0 = q_n$, on vérifie qu'une unité de temps est au moins écoulée dans ω en appliquant l'algorithme CompleteReach. L'état symbolique de départ S_0 est l'état (q_i, Z) avec $(x = 0) \subseteq Z$ et les états symboliques de la forme $(q_j, (Z_k) \cap (x \geq 1))$ avec $j \in \{0, \cdots, n-1\}$ comme ensemble d'états cibles $\mathcal{Z}_{\geq 1}$. Dans le cas du cycle formé par e_2, e_3 et e_4, on a $S_0 = (q_1, (x = y = 0))$ car e_2 remet x à zéro et pour états symboliques cibles $(q_1, Z_1 \cap (x \geq 1))$, $(q_2, Z_2 \cap (x \geq 1))$, $(q_2, Z_3 \cap (x \geq 1))$, $(q_3, Z_4 \cap (x \geq 1))$ et $(q_3, Z_5 \cap (x \geq 1))$. L'algorithme montre que $(q_2, Z_3 \cap (x \geq 1))$ est toujours accessible et donc qu'au moins une unité de temps s'est écoulée dans le cycle.

2.2 Un unique état final : l'état absorbant

2.2.1 Cas des processus de décision markoviens

On rappelle qu'un chemin ω dans une chaîne de Markov G [23, 12] est de la forme $\omega = q_0 \overset{p_0}{\rightarrow} q_1 \overset{p_1}{\rightarrow} q_2 \overset{p_2}{\rightarrow} \cdots \overset{p_{n-1}}{\rightarrow} q_n$.

Définition 2.7. *(Etat récurrent.) Soit $i \in \mathbb{N}$. On dit que l'état $\omega(i)$ est* récurrent *si la probabilité que la chaîne*[3]*, partant de $\omega(i)$, repasse par $\omega(i)$, est égale à 1. Dans le cas contraire, il est* transient*. Une chaîne de Markov est dite* récurrente *si tous ses états sont récurrents. Formellement, si P est la matrice de transitions probabilistes associée à la chaîne de Markov G, un état $\omega(i)$ est récurrent si et seulement si il existe $n \in \mathbb{N}$ tel que $P^{(n)}(\omega(i), \omega(i)) = 1$. La probabilité que la chaîne partant de $\omega(i)$ repasse par $\omega(i)$ après un nombre fini de transitions, est égale à 1.*

Définition 2.8. *(Etat accessible.) On dit qu'un état q' de G est* accessible *à partir d'un état q si la chaîne a une probabilité strictement positive de passer de q à q' i.e il existe un chemin ω de G avec $\omega = q \overset{p_{i_0}}{\rightarrow} q_{i_1} \overset{p_{i_1}}{\rightarrow} \cdots \overset{p_{i_j}}{\rightarrow} q'$ tel que $p_{i_k} > 0$ pour tout $0 \leq k \leq j$.*

On déduit que tout état accessible à partir d'un état récurrent est également récurrent et appartient à la même composante connexe. On peut ainsi définir les classes récurrentes de G.

Définition 2.9. *(Classe récurrente.) Un ensemble non vide d'états d'une chaîne de Markov G est appelé* classe récurrente *si tous les états de cet ensemble sont récurrents et appartiennent à la même composante fortement connexe.*

Définition 2.10. *(Ensemble clos.) On dit qu'un ensemble C d'états de G est* clos *si les seuls états accessibles à partir d'un état de C appartiennent à C.*

[3]La chaîne de Markov est supposée homogène [23, 12].

On définit les ensembles irréductibles à partir des ensembles clos.

Définition 2.11. *(Ensemble irréductible.) Un ensemble C est dit* irréductible *si aucun de ses sous-ensembles n'est clos.*

Si l'ensemble C se restreint à un unique élément, on dira que cet élément est *absorban*t.

Soit $\mathcal{M}$ un processus de décision markovien (PDM) [30]. Soit Adv l'ensemble des adversaires de $\mathcal{M}$.

Définition 2.12. *(PDM Unichaîne.) On dit que $\mathcal{M}$ est unichaîne si pour tout $A \in Adv$, il existe une unique classe récurrente avec une possibilité d'avoir un ensemble vide E d'états transients dans A.*

La définition de PDM unichaîne généralise la définition de PDM récurrente où l'ensemble E est toujours vide pour tout adversaire A du PDM.
Par ailleurs, pour vérifier qu'un PDM est unichaîne, on peut appliquer l'algorithme de Fox-Landi [30] à la chaîne de Markov correspondant à chaque adversaire du PDM. Cet algorithme calcule les classes récurrentes et les états transients de chaînes de Markov finies.

Définition 2.13. *(Etat absorbant.) Un état q d'un PDM $\mathcal{M}$ est dit* absorbant *si pour tout adversaire $A \in Adv$ et pour tout chemin $\omega \in Path^A_{fin}$ avec $last(\omega) = q$, on a $Prob^A_{fin}(\omega') = Prob^A_{fin}(\omega)$ avec $\omega' = \omega \xrightarrow{0,1} q$ et $\mathbf{P^A}(\omega, \omega') = \mathbf{1}$.*

Les états absorbants d'un PDM $\mathcal{M}$ correspondent à l'ensemble des états absorbants des adversaires de $\mathcal{M}$.

Considérons maintenant que pour tout adversaire A de $\mathcal{M}$, il existe un unique état absorbant q_A. Si l'on vérifie par l'algorithme de Fox-Landi que tout A ne possède qu'une seule classe récurrente (ou ensemble irréductible clos), alors cette classe sera égale à $\{q_A\}$. Les états de A excepté q_A sont donc transients. On peut donc atteindre l'état q_A à partir de tout autre état de A avec une probabilité égale à 1.
Soit q_{end} l'unique état absorbant pour tout adversaire A de $\mathcal{M}$. La seule classe récurrente de $\mathcal{M}$ sera q_{end} et les autres états de $\mathcal{M}$ seront transients.
On peut ainsi déduire ce qui suit pour tout PDM unichaîne :

$$Prob^A\{\omega | \omega \in Path^A_{ful} \wedge last(\omega) = q_{end}\} = 1$$

2.2.2 Cas des automates temporisés probabilistes

Pour un automate temporisé probabiliste $\mathcal{A}$, on considère le système temporisé probabiliste qui lui est associé $\mathcal{M}_\mathcal{A}$. De ce système temporisé probabiliste, on ne tiendra compte que des adversaires divergents de $\mathcal{A}$ dans l'ensemble $Adv^\mathcal{A}_{div}$.

D'autre part, une deuxième méthode peut être proposée. Elle consiste à construire l'automate des zones $\mathcal{Z}(\mathcal{A})$ associé à $\mathcal{A}$ (voir section 1.1.3). Toute transition (S, e, S') de cet automate des zones, où S et S' sont des états symboliques de $\mathcal{Z}(\mathcal{A})$ et $e = (q, g, p, X, q')$ une transition probabiliste de $\mathcal{A}$ est munie de la probabilité p. On considère ainsi $\mathcal{Z}(\mathcal{A})$ muni des distributions de $\mathcal{A}$ en chaque état symbolique $S = (q, Z)$.

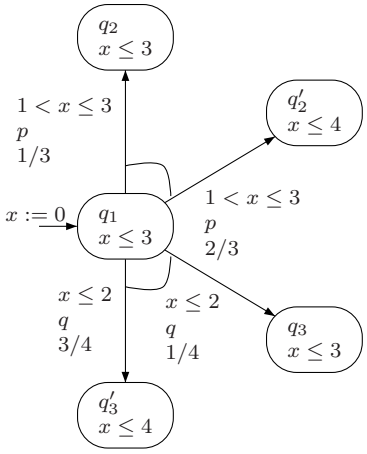

FIG. 2.1 – Figure de l'exemple 2.3

2.3 Déterminisme d'actions

2.3.1 Automate temporisé probabiliste déterministe

Soit $\mathcal{A}$ un automate temporisé probabiliste bien formé. Etant donnés un état $q \in Q$ et un ensemble de distributions $prob(q)$ associé à q, une garde $g = \tau_q(p)$ est au moins vérifiée à un instant $t \in \mathbb{R}^+$ pour une distribution $p \in prob(q)$. Dans le cas où il y a au moins deux gardes $g_i = \tau_q(p_i)$ vérifiées à $t \in \mathbb{R}^+$, nous parlerons de *non déterminisme d'actions*.

Exemple 2.3. *La figure 2.1 décrit une partie d'un automate temporisé probabiliste bien formé en q où le choix des actions peut se faire de manière non déterministe. Soit ν une valuation. L'horloge x étant remise à zéro en entrant dans l'état q_1 (i.e $\nu(x) = 0$), on peut y rester au plus 3 unités de temps.*
Si l'on laisse t unités de temps s'écouler de façon à avoir $\nu'(x) = \nu(x) + t \leq 1$, seules les transitions probabilistes engendrées par la distribution q sont tirables. Pour un adversaire A en q_1 à l'instant t, A peut laisser le temps s'écouler ou bien choisir de manière déterministe la distribution q.
Par contre, si l'on choisit de rester dans l'état q_1, t unités de temps telles que $1 < \nu'(x) \leq 2$, toutes les transitions probabilistes engendrées par les distributions p et q seront alors activées. Pour un adversaire A en q_1 à l'instant t, A peut laisser le temps s'écouler ou bien faire un choix non déterministe sur l'ensemble des distributions $\{p, q\}$.

Nous pouvons ainsi donner une définition des automates temporisés probabilistes bien formés déterministes comme suit :

Définition 2.14. *Un automate temporisé probabiliste bien formé $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, inv, prob, \langle \tau_q \rangle_{q \in Q})$ est déterministe si pour tout état $q \in Q$ et pour toute valuation ν tel que $\langle q, \nu \rangle$ est non bloquant, une seule garde $g = \tau_q(p)$ relative à une distribution $p \in prob(q)$ est vérifiée, i.e pour $t \in \mathbb{R}^+$, on a $\nu + t \models g$ et $\nu + t \not\models g'$ pour tout $g' = \tau_q(p')$ tel que $p' \in prob(q) \setminus p$.*

Ainsi, dans un automate temporisé probabiliste déterministe, seul un choix probabiliste sur les transitions engendrées par la distribution activée (q, g, p) est nécessaire pour passer à l'état suivant.

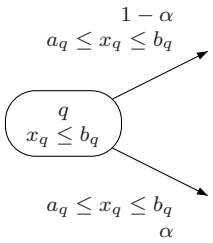

FIG. 2.2 – Exemple d'un état dans un ATPSD

2.3.1.1 Automate temporisé probabiliste paramétré trivialement déterministe

Par ailleurs, dans notre travail nous ferons uniquement appel à des automates temporisés probabilistes paramétrés (ATPP) (voir la définition 1.8) où une unique distribution est associée à chaque état $q \in Q$. Nous donnons ainsi la définition suivante :

Définition 2.15. *(Automate temporisé probabiliste paramétré trivialement déterministe.) Un ATPP* $\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, inv, prob, \langle \tau_q \rangle_{q \in Q})$ *est* trivialement déterministe *si la fonction prob attribue à chaque état* $q \in Q$ *une unique distribution* $prob(q)$ *sur* $2^{\mathcal{X}} \times Q$ *telle que* $prob : Q \rightarrow Dist(2^{\mathcal{X}} \times Q)$.

Remarque 2.2. *D'après la définition 1.14 des automates temporisés probabilistes, nous définissons les automates temporisés probabilistes trivialement déterministes de la même manière que la définition 2.15.*

2.4 Définition des Automates Temporisés Probabilistes Déterminés (ATPD)

2.4.1 Les Automates Temporisés Probabilistes Semi Déterminés (ATPSD)

On rappelle que l'ensemble $\mathcal{C}(\mathcal{X})$ désigne l'ensemble des contraintes temporelles d'un automate temporisé probabiliste sans contraintes diagonales de la forme $x - y \sim c$. Nous noterons dans ce qui suit $\mathcal{C}_{\leq}(\mathcal{X})$, $\mathcal{C}_{\geq}(\mathcal{X})$ et $\mathcal{C}_{=}(\mathcal{X})$ les ensembles de contraintes temporelles atomiques d'un automate temporisé probabiliste, respectivement de la forme $x \leq c$, $x \geq c$ et $x = c$ avec $c \in \mathbb{N}$.
Nous allons définir une sous classe d'automates temporisés probabilistes trivialement déterministes (voir section 2.3.1.1) appelée automates temporisés probabilistes semi déterminés (ATPSD). Dans un ATPSD, la garde qu'on attribue à l'unique distribution en chaque état ne fait intervenir qu'une seule horloge et se présente sous la forme d'une conjonction $g_1 \wedge g_2$ tel que $g_1 \in \mathcal{C}_{\leq}(\mathcal{X})$ et $g_2 \in \mathcal{C}_{\geq}(\mathcal{X})$. La figure 2.2 décrit la forme de la garde et de l'invariant associés à un état dans un ATPSD.

Nous donnons maintenant une définition précise des ATPSDs comme suit.

Définition 2.16. *(Automate Temporisé Probabiliste Semi Déterminé (ATPSD)). Un automate temporisé probabiliste trivialement déterministe* $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, \phi, \psi, inv, prob, \langle \tau_q \rangle_{q \in Q})$ *est* Semi Déterminé *(ATPSD) si :*

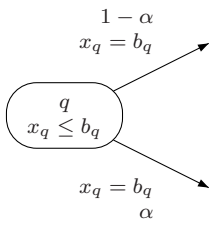

FIG. 2.3 – Exemple d'un état dans un ATPD

- $\psi : Q \to \mathcal{X}$ *est une fonction qui associe à chaque état* $q \in Q$ *une horloge.*
- $\phi : Q \to \mathbb{N}$ *est une fonction qui associe à chaque état* $q \in Q$ *un entier.*
- $inv : Q \to \mathcal{C}_{\leq}(\mathcal{X})$ *associe à chaque état un invariant de la forme* $\psi(s) \leq \phi(s) \in \mathcal{C}_{\leq}(\mathcal{X})$.
- *la famille de fonctions* $\langle \tau_q \rangle_{q \in Q}$ *associe à toute distribution* $prob(q)$ *une garde de la forme* $a_q \leq \psi(q) \leq \phi(q)$ *telle que* $a_q \leq \psi(q) \in \mathcal{C}_{\geq}(\mathcal{X})$ *pour un certain entier* a_q *positif ou nul et* $\psi(q) \leq \phi(q) \in \mathcal{C}_{\leq}(\mathcal{X})$.

Donnons un aperçu sur les caractéristiques d'un ATPSD.
A tout état $q \in Q$, on associe une horloge $\psi(q)$ sur laquelle portera l'unique contrainte temporelle atomique de $inv(q)$ (soit $\psi(q) \leq \phi(q)$ où $\phi(q)$ est un entier) et les contraintes de la garde $\tau_q(prob(q))$ associée à la distribution $prob(q)$ issue de q.

Nous nous intéresserons par la suite aux ATPSDs fortement non zenon (voir section 2.1.2), bien formé (voir section 2.1.1) et possédant un unique état final absorbant noté q_{end} comme cela est décrit dans la section 2.2. Toutes ces propriétés peuvent être vérifiées grâce aux algorithmes décrits dans les sections correspondantes.

Exemple 2.4. *L'automate temporisé probabiliste décrit dans l'exemple 1.1 est un ATPSD. Prenons par exemple l'état* q_3. *On a* $\psi(q_3) = z$, $\phi(q_3) = 2$ *et les transitions probabilistes* e_4 *et* e_5 *sont engendrées par l'unique distribution* $prob(q_3)$ *associée à* q_3. *La garde* $\tau_{q_3}(prob(q_3))$ *associée à la distribution* $prob(q_3)$ *est* $0 \leq z \leq 2$. *Nous avons déjà montré que cet automate était bien formé et fortement non zenon respectivement dans les exemples 2.1 et 2.2.*

2.4.2 Les Automates Temporisés Probabilistes Déterminés (ATPD)

2.4.2.1 Définition

Les automates temporisés probabilistes déterminés (ATPD) sont eux aussi une sous classe des automates temporisés probabilistes trivialement déterministes.
Dans un ATPD, la garde qu'on attribue à l'unique distribution en chaque état ne fait intervenir qu'une seule horloge et appartient à l'ensemble de contraintes temporelles atomiques $\mathcal{C}_{=}(\mathcal{X})$.
La figure 2.3 donne la forme de l'invariant et de la garde associés à un état dans un ATPD.

Définition 2.17. *(Automate Temporisé Probabiliste Déterminé (ATPD)). Un automate temporisé probabiliste trivialement déterministe* $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, \phi, \psi, inv, prob, \langle \tau_q \rangle_{q \in Q})$ *est* Déterminé *(ATPD) si :*

- $\psi : Q \to \mathcal{X}$ *est une fonction qui associe à chaque état* $q \in Q$ *une horloge.*
- $\phi : Q \to \mathbb{N}$ *est une fonction qui associe à chaque état* $q \in Q$ *un entier.*

- *$inv : Q \to \mathcal{C}_{\leq}(\mathcal{X})$ associe à chaque état un invariant de la forme $\psi(q) \leq \phi(q) \in \mathcal{C}_{\leq}(\mathcal{X})$.*
- *une famille de fonctions $\langle \tau_q \rangle_{q \in Q}$ qui associe à toute distribution $prob(q)$ une garde de la forme $(\psi(q) = \phi(q)) \in \mathcal{C}_{=}(\mathcal{X})$.*

Un ATPD est un cas particulier d'un ATPSD. En effet, un ATPD est un ATPSD dans lequel toutes les gardes du type $a_q \leq \psi(q) \leq \phi(q)$ sont des contraintes d'égalité de la forme $\psi(q) = \phi(q)$.

Remarque 2.3. *Un Automate $\mathcal{A}$ est un Automate Temporisé Probabiliste Déterminé (ATPD) si $\mathcal{A}$ est un Automate Temporisé Probabiliste Semi Déterminé dont la famille de fonction $\langle \tau_q \rangle_{q \in Q}$ associe à toute distribution $prob(q)$ une garde de la forme $(\psi(q) = \phi(q))$.*

Proposition 2.1. *Soit $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, \phi, \psi, prob, \langle \tau_q \rangle_{q \in Q})$ un ATPD bien formé. Soient $q \in Q$ et $prob(q)$ l'unique distribution probabiliste issue de q. Soit ν une valuation telle que $\langle q, \nu \rangle$ est accessible. Alors il existe un unique $t \in \mathbb{R}^+$ tel que toute transition probabiliste de la forme $(q, \psi(q) = \phi(q), prob(q)(q'), q')$ issue de q devient tirable au bout de t unités de temps. Ce temps est égal à $\phi(q) - \nu(\psi(q))$.*

Démonstration: Comme l'automate est supposé bien formé, il existe $t \in \mathbb{R}^+$ tel que la transition $(q, \psi(q) = \phi(q), prob(q)(q'), q')$ est tirable au temps t avec $\langle q, \nu \rangle$ accessible. En particulier, $\nu + t$ vérifie la garde $(\psi(q) = \phi(q))$ avec $\nu + t \models (\psi(q) = \phi(q))$, soit $\nu(\psi(q)) + t = \phi(q)$. Ainsi, $t = \phi(q) - \nu(\psi(q))$ est déterminé de façon unique et ne dépend pas de q'. □

Notation 2.1. *Le temps écoulé en q égal à $\phi(q) - \nu(\psi(q))$ est noté $Exit(q, \nu)$.*

Corollaire 2.1. *Avec les notations de la proposition 2.1, pour tout chemin [4] ω de l'ATPD $\mathcal{A}$,*

$$\omega = \langle q_0, \nu_0 \rangle \xrightarrow[t_0]{e_0} \langle q_1, \nu_1 \rangle \xrightarrow[t_1]{e_1} \cdots \xrightarrow[t_{n-1}]{e_{n-1}} \langle q_n, \nu_n \rangle,$$

$t_{n-1} = Exit(q_{n-1}, \nu_{n-1})$ est uniquement déterminé par ν_0 et les transitions probabilistes e_j avec $j \in \{0, \cdots, n-2\}$. En particulier, la durée de ω est uniquement déterminée par ν_0 et est égale à $\sum_{i=0}^{n-1} Exit(q_i, \nu_i)$.

Démonstration: Notons $e_i = (q_i, (\psi(q_i) = \phi(q_i)), X_i, q_{i+1})$ pour tout $i \in \{0, \cdots, n-1\}$.

On a $t_0 = Exit(q_0, \nu_0)$ et ainsi $\nu_1 = (\nu_0 + t_0)[X_0 := 0]$. $Exit(q_1, \nu_1)$ est donc déterminé par ν_0. Par récurrence sur i, supposons que pour tout $j \leq i$, t_j est uniquement déterminé par ν_0. Ainsi, ν_{i+1} est uniquement déterminé par ν_0. Par la propostion 2.1, t_{i+1} est déterminé par ν_{i+1} donc par ν_0 car $t_{i+1} = Exit(q_{i+1}, \nu_{i+1})$. □

Un ATPD $\mathcal{A}$ est donc totalement déterministe dans le sens où il n'y a pas de non déterminisme de temps. Toute transition probabiliste ne peut être tirée que lorsque l'horloge associée à l'état atteint une valeur fixée et que le temps écoulé en cet état est déterminé par rapport à la valuation initiale ν_0. En particulier, un ATPD $\mathcal{A}$ est une chaîne de Markov à coûts (voir section 1.6.1) représentés par le temps. Si l'on supprime les données temporelles dans un ATPD $\mathcal{A}$ on obtient une chaîne de Markov (voir la section 1.2) notée $Untimed(\mathcal{A})$.

[4] Dans un ATPD, les termes *chemin* et *exécution* sont équivalents car le temps est fixé. Il existe une unique exécution sur un chemin donné.

2.4.2.2 ATPD associé à un ATPSD

Définition 2.18. *Un ATPSD* $\mathcal{A} = (Q, \bar{q}, \mathcal{X}, \phi, \psi, inv, prob, \langle \tau_q \rangle_{q \in Q})$ *définit un ATPD, noté* $ATPD(\mathcal{A})$*, de la forme* $ATPD(\mathcal{A}) = (Q, \bar{q}, \mathcal{X}, \phi, \psi, inv, prob, \langle \tau'_q \rangle_{q \in Q})$ *avec pour tout état* $q \in Q$ $\tau'_q(prob(q)) = (\psi(q) = \phi(q))$ *si* $\tau_q(prob(q)) = (a_q \leq \psi(q) \leq \phi(q))$*.*

Remarque 2.4. *Toute exécution dans* $ATPD(\mathcal{A})$ *est une exécution dans* $\mathcal{A}$*.*

Lemme 2.3. *Soit* $\mathcal{A}$ *un ATPSD et* $\langle q, \nu \rangle$ *un état accessible dans* $ATPD(\mathcal{A})$*. Soit* $\langle q, \nu \rangle \xrightarrow[t]{e} \langle q', \nu' \rangle$ *une transition dans* $\mathcal{A}$*. Alors il existe un unique* $t' \in \mathbb{R}^+$*, tel que* $\langle q, \nu \rangle \xrightarrow[t']{e} \langle q', \nu'' \rangle$ *est une transition dans* $ATPD(\mathcal{A})$ *et* $t' \geq t$*.*

Démonstration: On pose $t' = Exit(q_1, \nu_1)$ d'après la proposition 2.1. Pour tout $t \in \mathbb{R}^+$, tel que $\langle q, \nu \rangle \xrightarrow[t]{e} \langle q', \nu' \rangle$ est une transition dans $\mathcal{A}$, on a $t \leq \phi(q_1) - \nu(\psi(q_1))$ avec $\phi(q_1) - \nu(\psi(q_1)) = Exit(q_1, \nu_1)$. Pour $t = \phi(q_1) - \nu(\psi(q_1))$, $\langle q, \nu \rangle \xrightarrow[t]{e} \langle q', \nu'' \rangle$ est une transition dans $ATPD(\mathcal{A})$. □

Par le lemme 2.3, on déduit la proposition suivante :

Proposition 2.2. *Soit* $\mathcal{A}$ *un ATPSD. Soit*

$$\omega = \langle q_0, \nu_0 \rangle \xrightarrow[t_0]{e_0} \langle q_1, \nu_1 \rangle \xrightarrow[t_1]{e_1} \cdots \xrightarrow[t_{n-1}]{e_{n-1}} \langle q_n, \nu_n \rangle,$$

une exécution dans $\mathcal{A}$*. Alors il existe une unique exécution*

$$\omega' = \langle q_0, \nu_0 \rangle \xrightarrow[t'_0]{e'_0} \langle q_1, \nu_1 \rangle \xrightarrow[t'_1]{e'_1} \cdots \xrightarrow[t'_{n-1}]{e'_{n-1}} \langle q_n, \nu_n \rangle,$$

dans $ATPD(\mathcal{A})$ *qui vérifie* $\forall\ i \in \{0, \cdots, n-1\}$, $t_i{}' \geq t_i$*.*

Proposition 2.3. *Si l'ATPSD* $\mathcal{A}$ *est bien formé (resp. fortement non zenon), alors* $ATPD(\mathcal{A})$ *est bien formé (resp. fortement non zenon).*

Démonstration: Supposons $\mathcal{A}$ bien formé et montrons que $ATPD(\mathcal{A})$ est bien formé :

Soient q un état et ν une valuation telle que $\langle q, \nu \rangle$ est accessible dans $ATPD(\mathcal{A})$ (donc dans $\mathcal{A}$) et $\nu \models inv(q)$, soit $\nu(\psi(q)) \leq \phi(q)$. Soit $\tau \in \mathbb{R}^+$ tel que $\nu(\psi(q)) + \tau = \phi(q)$. Alors $\nu + \tau$ est une valuation qui satisfait $inv(q)$. Comme $\mathcal{A}$ est bien formé, il existe une transition probabiliste $e = (q, (\psi(q) \leq \phi(q)), prob(q)(q'), X, q')$ issue de q, tirable à partir de $\langle q, \nu + \tau \rangle$. En particulier, $\nu' = (\nu + \tau)[X := 0] \models inv(q')$ et donc la transition $e = (q, \psi(q) = \phi(q), prob(q)(q'), X, q')$ de l'ATPD $ATPD(\mathcal{A})$ est tirable de $\langle q, \nu + \tau \rangle$. Ainsi, on a montré que $ATPD(\mathcal{A})$ est bien formé.

Supposons que $\mathcal{A}$ est fortement non zenon et montrons que $ATPD(\mathcal{A})$ est fortement non zenon :

Soit

$$\langle q_0, \nu_0 \rangle \xrightarrow{e_0} \langle q_1, \nu_1 \rangle \xrightarrow{e_1} \cdots \xrightarrow{e_{n-1}} \langle q_n, \nu_n \rangle$$

un cycle de $ATPD(\mathcal{A})$ avec $q_0 = q_n$. Il lui correspond un cycle dans $\mathcal{A}$

$$\langle q_0, \nu_0 \rangle \xrightarrow{\hat{e}_0} \langle q_1, \nu_1 \rangle \xrightarrow{\hat{e}_1} \cdots \xrightarrow{\hat{e}_{n-1}} \langle q_n, \nu_n \rangle$$

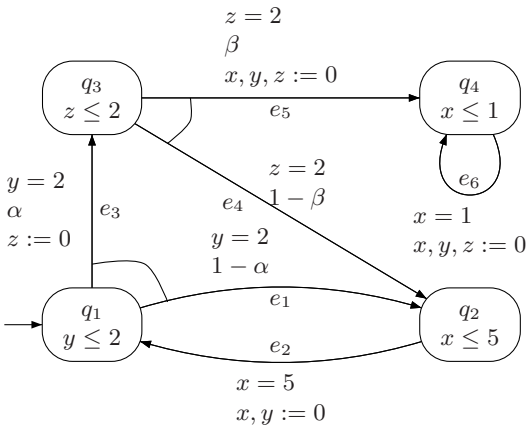

FIG. 2.4 – ATPD de l'exemple 2.5

où $e_i = (q_i, \psi(q_i) = \phi(q_i), prob(q_i)(q_{i+1}), X_i, q_{i+1})$ et $\hat{e}_i = (q_i, \psi(q_i) \leq \phi(q_i), prob(q_i)(q_{i+1}), X_i, q_{i+1})$.

Comme $\mathcal{A}$ est fortement non zenon, il existe une horloge $x \in \mathcal{X}$ et $1 \leq i, j \leq m$ telle que l'horloge $\psi(q_i) = x$ est remise à zéro par la transition probabiliste $\hat{e}_i$ et $\phi(q_j) \geq 1$. Donc il existe une horloge $x \in \mathcal{X}$ et $1 \leq i, j \leq m$ telle que l'horloge $\psi(q_i) = x$ est remise à zéro par la transition probabiliste e_i et $\phi(q_j) \geq 1$ dans $ATPD(\mathcal{A})$. Ainsi, on a montré que $ATPD(\mathcal{A})$ est fortement non zenon.

□

Exemple 2.5. *Considérons l'ATPD décrit dans la figure 2.4. Cet ATPD correspond à l'ATPSD de l'exemple 1.1. Le chemin ω considéré dans l'exemple 1.1 est décrit de la manière suivante :*

$$\omega = \langle q_1, \nu_1 \rangle \xrightarrow{2} \langle q_3, \nu_2 \rangle \xrightarrow{2} \langle q_2, \nu_3 \rangle \xrightarrow{1} \langle q_1, \nu_4 \rangle.$$

La durée écoulée dans chaque état q_i de ce chemin est déterminée par rapport à la valuation ν_1. La durée totale de ce chemin est par conséquent fixée et est égale à : $(2 - \nu_1(\psi(q_1))) + (2 - \nu_2(\psi(q_3))) + (1 - \nu_3(\psi(q_2)))$. Or les valuations ν_2 et ν_3 sont fonctions de la valuation ν_1 et donc de la durée écoulée en q_1, égale à $(2 - \nu_1(\psi(q_1)))$.

2.4.2.3 Le pire temps moyen de convergence dans un ATPSD

Soit $\mathcal{A}$ un ATPSD absorbant. On pose q_{end} l'unique état final absorbant de $\mathcal{A}$.
On souhaite calculer le pire temps moyen de convergence (i.e temps moyen de convergence maximal) vers l'état q_{end} dans un ATPSD $\mathcal{A}$ en partant de l'état initial q_0 avec une valuation ν_0 (i.e la valuation $\mathbf{0}$) remettant toutes les horloges de l'automate à zéro.

On déduit de la proposition 2.2, la proposition suivante :

Proposition 2.4. *Etant donné un ATPSD $\mathcal{A}$, le pire temps moyen de convergence est capturé par $ATPD(\mathcal{A})$.*

Ce comportement est par conséquent capturé par l'ATPD associé à cet ATPSD dont la garde associée à toute distribution $prob(q)$ est égale à $(\psi(q) = \phi(q))$.

Définition 2.19. *(Temps moyen de convergence dans un ATPD). Soit $\Omega(q_{end})$ l'ensemble des chemins d'un automate temporisé probabiliste déterminé absorbant tel que pour tout chemin $\omega \in \Omega(q_{end})$ de $\mathcal{A}$, on a $last(\omega) = q_{end}$ et il n'existe aucun préfixe ω' de ω tel que $last(\omega') = q_{end}$. On définit le temps moyen de convergence d'un ATPD vers l'état absorbant q_{end} par :*

$$ExpAbs(\mathcal{A}) = \sum_{\omega \in \Omega_{q_{end}}} Pr(\omega) Dur(\omega).$$

2.5 Les Automates Temporisés Probabilistes Paramétrés et Déterminés (ATPPD)

Dans les Automates Temporisés Probabilistes Déterminés décrits dans la section 2.4.2, à tout état $q \in Q$ la fonction ϕ associe un entier dans $\mathbb{N}$ à l'horloge $\psi(q)$.

On rappelle que l'ensemble $\mathcal{C}_{\mathcal{P}}(\mathcal{X})$ désigne l'ensemble des contraintes temporelles d'un automate temporisé probabiliste paramétré sans contraintes diagonales de la forme $x - y \sim c$. Nous noterons dans ce qui suit $\mathcal{C}_{\mathcal{P},\leq}(\mathcal{X})$, $\mathcal{C}_{\mathcal{P},\geq}(\mathcal{X})$ et $\mathcal{C}_{\mathcal{P},=}(\mathcal{X})$ les ensembles de contraintes temporelles atomiques d'un automate temporisé probabiliste paramétré, respectivement de la forme $x \leq c$, $x \geq c$ et $x = c$ avec $c \in \mathbb{N}$ ou $c \in \mathcal{P}$.
Nous présentons dans la suite les Automates Temporisés Probabilistes Paramétrés Déterminés qui forment une sous classe d'automates temporisés probabilistes paramétrés trivialement déterministes. Ils sont une généralisation des Automates Temporisés Probabilistes Déterminés (ATPD), dans le sens où les contraintes temporelles considérées seront dans $\mathcal{C}_{\mathcal{P}}(\mathcal{X})$ plutôt que dans $\mathcal{C}(\mathcal{X})$.
Nous avons vu que les comportements de blocage et fortement non zenon pouvaient être vérifiés et détectés respectivement par les algorithmes *Reach* et *TimelockReach* développés dans les sections 2.1.1 et 2.1.2 sur les automates temporisés probabilistes. Ces deux algorithmes font appel au graphe (ou automate) des zones de l'automate temporisé sous jacent à l'automate temporisé probabiliste considéré.
Par ailleurs, le graphe des zones est constitué d'un ensemble de polyèdres (convexes ou non convexes), faisant intervenir les contraintes temporelles atomiques non diagonales de l'automate en question. Ces contraintes sont de la forme $x \leq c$ dans le cas d'un ATPSD où c est un entier naturel. Nous verrons comment vérifier ces comportements dans le cas paramétré.
Nous introduisons tout d'abord les Automates Temporisés Probabilistes Paramétrés Semi Déterminés (ATPPSD) et ensuite les Automates Temporisés Probabilistes Paramétrés Déterminés (ATPPD).

2.5.1 Les Automates Temporisés Probabilistes Paramétrés Semi Déterminés (ATPPSD)

2.5.1.1 Définition

Nous définissons une sous classe d'automates temporisés probabilistes paramétrés trivialement déterministes (voir section 2.3.1.1) appelée Automates Temporisés Probabilistes Paramétrés Semi Déterminés (ATPPSD). Dans un ATPPSD, la garde qu'on attribue à l'unique distribution en chaque état de l'automate, ne fait intervenir qu'une seule horloge et se présente sous la forme d'une conjonction $g_1 \wedge g_2$ tel que $g_1 \in \mathcal{C}_{\mathcal{P},\leq}(\mathcal{X})$ et $g_2 \in \mathcal{C}_{\mathcal{P},\geq}(\mathcal{X})$.

Définition 2.20. *(Automate Temporisé Probabiliste Paramétré Semi Déterminé (ATPPSD).)*
Un automate temporisé probabiliste paramétré trivialement déterministe $\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, \psi, \phi, inv, prob, \langle \tau_q \rangle_{q \in Q})$ *est* Semi Déterminé *(ATPPSD) si :*

- $\psi : Q \to \mathcal{X}$ *est une fonction qui associe à chaque état* $q \in Q$ *une horloge.*
- $\phi : Q \to \mathcal{P} \cup \mathbb{N}$ *est une fonction qui associe à chaque état* $q \in Q$ *un paramètre* $a \in \mathcal{P}$ *ou un entier* $b \in \mathbb{N}$ *;*
- $inv : Q \to \mathcal{C}_{\mathcal{P},\leq}(\mathcal{X})$ *associe à chaque état un invariant de la forme* $(\psi(q) \leq \phi(q)) \in \mathcal{C}_{\mathcal{P},\leq}(\mathcal{X})$.
- *une famille de fonction* $\langle \tau_q \rangle_{q \in Q}$ *qui associe à toute distribution* $prob(q)$ *une garde de la forme* $a_q \leq \psi(q) \leq \phi(q)$ *où* $(a_q \leq \psi(q)) \in \mathcal{C}_{\mathcal{P},\geq}(\mathcal{X})$ *et* $(\psi(q) \leq \phi(q)) \in \mathcal{C}_{\mathcal{P},\leq}(\mathcal{X})$.

Soit $\mathcal{A}$ un ATPPSD et $\mathcal{P}$ l'ensemble de paramètres qui lui est associé. Pour toute valuation paramétrique $\kappa : \mathcal{P} \to \mathbb{N}$ qui associe à chaque paramètre a de $\mathcal{P}$ un entier dans $\mathbb{N}$, on obtient un ATPSD $\mathcal{A}_\kappa$ où chaque paramètre est remplacé par une valeur dans $\mathbb{N}$. L'ATPPSD $\mathcal{A}$ représente ainsi un ensemble $\mathcal{K}$ d'ATPSDs tel que $\mathcal{K} = \{\mathcal{A}_\kappa \mid \kappa \textit{ est une valuation paramétrique}\}$.

Exemple 2.6. *Nous allons donner un exemple d'une modélisation simplifiée du Bounded Retransmission Protocol (BRP) [15, 14] sous forme d'automate temporisé paramétré. L'ensemble des paramètres est de cardinal 2, soient* $\{T1, TD\}$*. L'automate final de la figure 2.9 est obtenu par composition synchronisée [32, 25] de 3 composants : un émetteur S de la figure 2.5, un récepteur R de la figure 2.6, un canal K de la figure 2.7 et un canal L de la figure 2.8. S émet un message à travers le canal K. Deux cas se profilent : le message est perdu ou bien le message parvient au récepteur. Si le récepteur reçoit le message, il émet à son tour un accusé de réception qui traverse le canal L. Si cet accusé de réception parvient à l'émetteur, ce dernier émet un nouveau message. La durée maximale de traversée des canaux K ou L est égale à* TD *unités de temps. Donc le temps maximal pour que l'accusé de réception parvienne à l'émetteur est* $2 \times TD$*. Si après* $2 \times TD$ *unités de temps, l'accusé de réception ne parvient pas à l'émetteur, ce dernier renvoie le même message*[5] *sinon il en envoie un nouveau après* $T1$ *unités de temps. Ainsi, dans cet exemple, nous avons les deux paramètres* $T1$ *et* TD *auxquels nous imposons la contrainte* $T1 > 2 \times TD$ *pour répondre à la spécification du protocole BRP. Restreignons ainsi le sous ensemble* $\Delta \subseteq \mathbb{T}^{\mathcal{P}}$ *avec* $\mathcal{P} = \{T1, TD\}$ *des valuations paramétriques à considérer. Toute valuation paramétrique* $\kappa \in \Delta$ *doit vérifier la condition* $\kappa(T1) > 2 \times \kappa(TD)$*. Une valuation paramétrique* κ *définie sur* $\mathcal{P}$ *par* $\kappa(T1) = 5$ *et* $\kappa(TD) = 2$ *appartient ainsi à* Δ*. En posant ces contraintes sur l'automate décrit dans la figure 2.9 on obtient l'ATPPSD décrit dans la figure 2.10, car le non déterminisme d'actions en* q_1 *et* q_3 *est ainsi supprimé. Le chemin* $q_1 \to q_1{}' \to q_3$ *de l'automate de la figure 2.9 est représenté par la transition* $q_1 \xrightarrow{e_3} q_3$ *dans l'automate de la figure 2.10. En effet, comme l'état* $q_1{}'$ *est transient (on y reste 0 unité de temps car* $t \leq 0$*), on le fusionne avec l'état* q_1 *qui devient ainsi état initial de l'automate à la place de* $q_1{}'$*. Une boucle de durée une unité de temps est ajoutée en* q_4 *pour éviter les comportements zenon.*

2.5.1.2 ATPPSD bien formé

Soit un ATPPSD $\mathcal{A}$ muni d'une valuation paramétrique κ. Si l'on remplace tout paramètre $a \in \mathcal{P}$ par $\kappa(a)$, on obtient un ATPSD noté $\mathcal{A}_\kappa$.

[5] Dans la modélisation complète du BRP [15, 14], le nombre de réémissions successives d'un même message est borné.

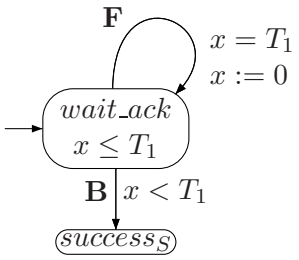

FIG. 2.5 – L'émetteur

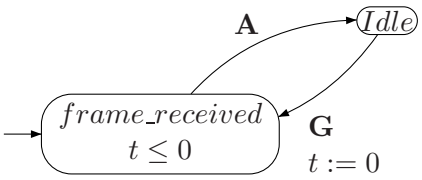

FIG. 2.6 – Le récepteur

Définition 2.21. *Soit $\mathcal{A}$ un ATPPSD. Soit Δ un sous ensemble de $\mathbb{T}^{\mathcal{P}}$. Nous dirons que $\mathcal{A}$ est bien formé pour Δ si pour toute valuation paramétrique $\kappa \in \Delta$, $\mathcal{A}_\kappa$ est un ATPSD bien formé.*

Remarque 2.5. *L'ensemble Δ étant possiblement infini, il n'est pas question de vérifier pour chaque $\mathcal{A}_\kappa$, $\kappa \in \Delta$, la propriété d'être bien formé ou non.*

Il faut donc déterminer un sous ensemble $\Delta \subseteq \mathbb{T}^{\mathcal{P}}$ pour lequel la propriété d'automates temporisés bien formés est vérifiée. En appliquant l'algorithme d'accessibilité des automates temporisés paramétrés comme cela est décrit dans [3, 20], on peut déterminer un ensemble Δ de valuations paramétriques qui permettent d'obtenir un ATPPSD bien formé, sauf que cet algorithme pourrait ne pas terminer car le problème d'accessibilité est indécidable dans le cas général d'après le théorème 1.1 (voir la section 1.4.2).

Il faut tout d'abord transformer l'ATPPSD en un automate temporisé paramétré en remplaçant le choix probabiliste en chaque état $q \in Q$ par un choix non déterministe et un état puit q_{sink} est créé. Toute transition sera dédoublée suivant la description ci-dessous.

Soit q un état de l'ATPPSD $\mathcal{A}$. Supposons par exemple que l'unique distribution $prob(q)$ en q engendre deux transitions probabilistes $e_1 = (q, \psi(q) \leq \phi(q), prob(q)(q'), X, q')$ et $e_2 = (q, \psi(q) \leq \phi(q), prob(q)(q''), Y, q'')$. Pour appliquer l'algorithme d'accessibilité, le choix probabiliste en q est remplacé par un choix non déterministe. Ainsi, e_1 est remplacée par $(q, \psi(q) \leq \phi(q), X, q')$ et e_2 par $(q, \psi(q) \leq \phi(q), Y, q'')$. Chaque nouvelle transition est ensuite représentée par deux transitions :

– Pour e_1 : ${e_1}' = (q, \psi(q) \leq \phi(q) \wedge \psi(q') \leq \phi(q'), X, q')$ et

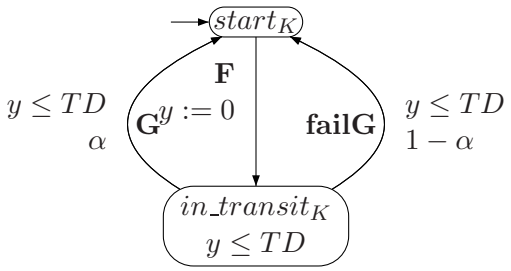

FIG. 2.7 – Canal K

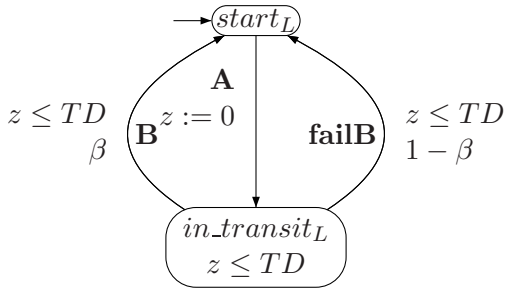

FIG. 2.8 – Canal L

$e_1{}'' = (q, \psi(q) \leq \phi(q) \wedge \psi(q') > \phi(q'), X, q_{sink})$.
– Pour e_2 : $e_2{}' = (q, \psi(q) \leq \phi(q) \wedge \psi(q'') \leq \phi(q''), X, q')$ et
$e_2{}'' = (q, \psi(q) \leq \phi(q) \wedge \psi(q'') > \phi(q''), X, q_{sink})$.

Exemple 2.7. *Reprenons l'exemple de l'ATPPSD du protocole BRP de la figure 2.10. Prenons par exemple la distribution prob(q_1). Elle engendre deux transitions probabilistes e_1 et e_3. La représentation de ces deux transitions dans l'automate temporisé paramétré équivalent à l'ATPPSD est représentée dans la figure 2.11.*

Les transitions $e_1{}''$ et $e_2{}''$ permettent d'avoir des relations entre les valeurs des horloges $\psi(q)$ et $\psi(q')$ d'une part et entre $\psi(q)$ et $\psi(q'')$ d'autre part. En effet, la garde $\psi(q) \leq \phi(q) \wedge \psi(q') > \phi(q')$ de la transition $e_1{}''$ est équivalente à la garde diagonale $\psi(q) - \psi(q') < \phi(q) - \phi(q')$. Il en est de même pour la transition $e_2{}''$ avec la garde diagonale $\psi(q) - \psi(q'') < \phi(q) - \phi(q'')$. L'état q n'est pas un état bloquant dans l'ATPPSD si les deux gardes $\psi(q) - \psi(q') < \phi(q) - \phi(q')$ et $\psi(q) - \psi(q'') < \phi(q) - \phi(q'')$ ne sont pas vérifiées en q. Ainsi, s'il existe une valuation ν telle que $\langle q, \nu \rangle$ soit accessible et $0 \leq \nu(\psi(q) - \psi(q')) < \kappa(\phi(q)) - \kappa(\phi(q'))$ (ou $0 \leq \nu(\psi(q) - \psi(q'')) < \kappa(\phi(q)) - \kappa(\phi(q''))$) pour $\kappa \in \mathbb{N}^{\mathcal{P}}$, alors l'état q_{sink} est accessible et $\kappa \notin \Delta$. Dans ce cas, l'état q est bloquant dans l'ATPPSD qui ne sera donc pas bien formé pour la valuation paramétrique κ.

Théorème 2.1. *Un ATPPSD $\mathcal{A}$ est bien formé si et seulement si l'état q_{sink} de l'automate temporisé paramétré équivalent à $\mathcal{A}$ n'est pas accessible.*

On rappelle que tout état accessible $\langle q, \nu \rangle$ d'un ATPSD n'est pas bloquant si $\nu \in free(q)$ comme nous l'avons vu dans la section 2.1.1.

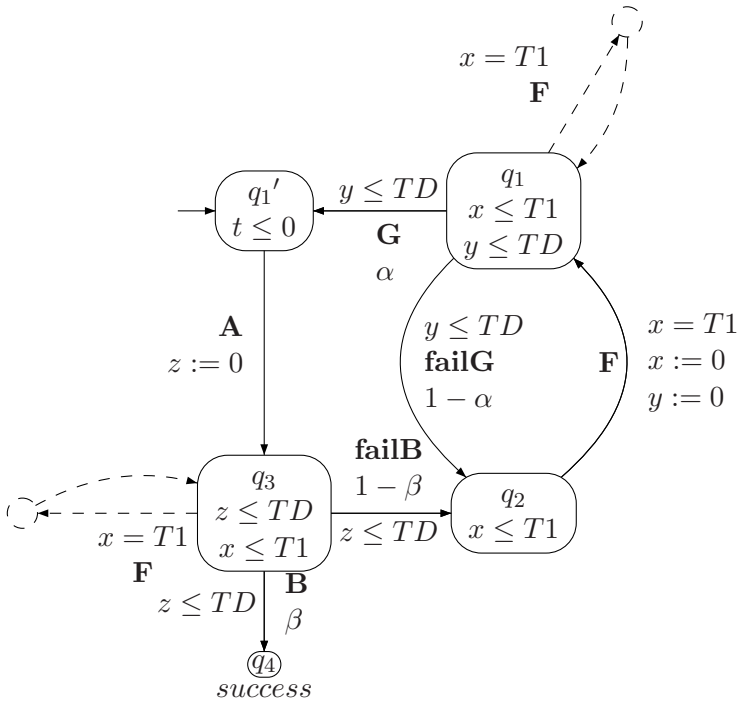

FIG. 2.9 – Produit des composants du protocole BRP sans la contrainte $T1 > 2TD$

Dans le cas d'un ATPPSD $\mathcal{A}$, pour tout $\kappa \in \Delta$, on définit $free_\kappa(q)$ pour tout état q de $\mathcal{A}_\kappa$. Pour tout état $\langle q, \nu \rangle$ accessible dans $\mathcal{A}_\kappa$, si $\nu \in free_\kappa(q)$, alors $\langle q, \nu \rangle$ n'est pas bloquant dans $\mathcal{A}_\kappa$.

D'autre part, pour déterminer l'ensemble de valuations paramétriques Δ cohérent avec l'ATPPSD pour la propriété d'automates bien formés, on cherche à l'aide de l'outil HyTech les valeurs sûres (voir la définition 1.13 de la section 1.4.2) pour tout paramètre dans $\mathcal{P}$ apparaissant dans l'automate temporisé paramétré. De ce fait, on cherche un ensemble de valuations paramétriques cohérentes avec la propriété d'accessibilité de l'état q_{sink}. L'automate temporisé paramétré soumis à l'outil HyTech est l'automate temporisé paramétré dérivé de l'automate temporisé probabiliste, en appliquant la transformation décrite précédemment.

Exemple 2.8. *Reprenons l'ATPPSD de l'exemple 2.6 représenté par la figure 2.10. Nous avons déjà imposé des contraintes sur le choix des valuations paramétriques pour répondre aux spécifications du protocole BRP. L'ensemble Δ est fixé tel que pour tout $\kappa \in \Delta$, on a $\kappa(T1) > 2 \times \kappa(TD)$. Par ailleurs, si nous voulons chercher un ensemble Δ' tel que l'automate initial décrit par la figure 2.9 soit bien formé, une étude en chaque état de l'automate est faite en utilisant les ensembles $free_\kappa(q)$ avec $\kappa \in \mathbb{N}^{\mathcal{P}}$. On constate qu'aucune exécution de l'automate n'atteint l'état q_{sink} quelle que soit la valuation paramétrique κ. L'automate de la figure 2.9 est donc bien formé pour tout $\kappa \in \mathbb{N}^{\mathcal{P}}$. Nous pouvons ainsi dire que l'automate de la figure 2.9 est* mathématiquement *bien formé pour tout $\kappa \in \mathbb{N}^{\mathcal{P}}$. Une contrainte* physique *du protocole avec $T1 > 2 \times TD$ est cependant imposée et permet de supprimer le non déterminisme d'actions*

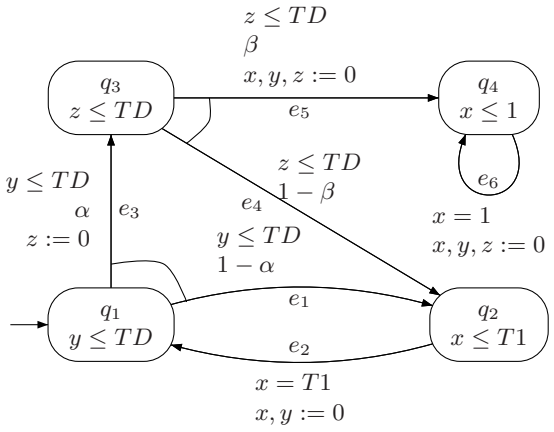

FIG. 2.10 – ATPPSD du protocole BRP

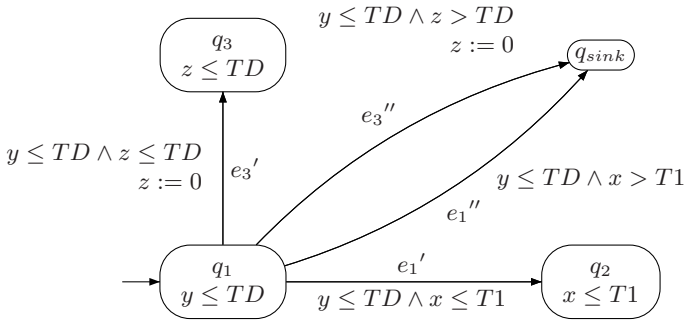

FIG. 2.11 – Exemple de transformation de la distribution $prob(q_1)$ de l'ATPPSD du protocole BRP

dans les états q_1 et q_3. Pour l'ATPPSD associé au protocole BRP, on considère finalement l'ensemble Δ.

2.5.1.3 ATPPSD fortement non zenon

Après avoir déterminé un ensemble Δ de valuations paramétriques qui permet d'éviter tout état de blocage dans l'ATPPSD $\mathcal{A}$ pour tout $\kappa \in \Delta$, il faut vérifier que cet automate est fortement non zenon. Cette propriété se vérifie de la façon décrite dans la section 2.1.2, utilisant l'algorithme $TimelockReach$ qui permet de détecter l'existence de comportements zenon en général. On rappelle que dans le cas de la propriété fortement non zenon, on s'intéresse à l'écoulement d'au moins une unité de temps dans tout cycle de l'automate temporisé. Pour une valuation paramétrique $\kappa \in \Delta$ donnée, on peut ainsi appliquer $TimelockReach$ sur tout cycle de l'ATPPSD muni de κ. Cette vérification se fait par conséquent sur l'ensemble des ATPSD $\mathcal{A}_\kappa$.

Un ATPPSD $\mathcal{A}$ est fortement non zenon par rapport à une valuation paramétrique $\kappa \in \Delta$, si pour tout cycle $\langle q_0, \nu_0 \rangle \xrightarrow{e_0}_\kappa \langle q_1, \nu_1 \rangle \xrightarrow{e_1}_\kappa \cdots \xrightarrow{e_{n-1}}_\kappa \langle q_n, \nu_n \rangle$ tel que $q_0 = q_n$, il existe une horloge $x \in \mathcal{X}$ et $1 \leq i, j \leq m$ telle que l'horloge $\psi(q_i) = x$ est remise à zéro par la transition probabiliste e_i et $\kappa(\phi(q_j)) \geq 1$.

Exemple 2.9. *Prenons l'ATPPSD $\mathcal{A}$ décrit par la figure 2.10. L'ATPSD de l'exemple 2.4 est équivalent à cet ATPPSD muni de la valuation paramétrique κ avec $\kappa(T1) = 5$ et $\kappa(TD) = 2$. Le temps écoulé dans toute exécution ω de l'ATPPSD est déterminé par κ et la valuation initiale ν_0.*

2.5.2 Les Automates Temporisés Probabilistes Paramétrés Déterminés (ATPPD)

2.5.2.1 Définition

Les Automates Temporisés Probabilistes Paramétrés Déterminés (ATPPD) sont eux aussi une sous classe des automates temporisés probabilistes paramétrés trivialement déterministes. Dans un ATPPD, la garde qu'on attribue à l'unique distribution en chaque état ne fait intervenir qu'une seule horloge et appartient à l'ensemble de contraintes temporelles atomiques $\mathcal{C}_{\mathcal{P},=}(\mathcal{X})$.

Définition 2.22. *(Automate Temporisé Probabiliste Paramétré Déterminé (ATPPD).) Un automate temporisé probabiliste paramétré trivialement déterministe $\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, \phi, \psi, inv, prob, \langle\tau_q\rangle_{q\in Q})$ est* Déterminé *(ATPPD) si :*

- *$\psi : Q \to \mathcal{X}$ est une fonction qui associe à chaque état $q \in Q$ une horloge.*
- *$\phi : Q \to \mathcal{P} \cup \mathbb{N}$ est une fonction qui associe à chaque état $q \in Q$ un paramètre $a \in \mathcal{P}$ ou un entier $b \in \mathbb{N}$;*
- *la fonction $inv : Q \to \mathcal{C}_{\mathcal{P},\leq}(\mathcal{X})$ associe à chaque état un invariant de la forme $\psi(q) \leq \phi(q) \in \mathcal{C}_{\mathcal{P},\leq}(\mathcal{X})$.*
- *une famille de fonction $\langle\tau_q\rangle_{q\in Q}$ qui associe à toute distribution $prob(q)$ une garde de la forme $(\psi(q) = \phi(q)) \in \mathcal{C}_{\mathcal{P},=}(\mathcal{X})$*

Notation 2.2. *On note $Path(\mathcal{A}, \kappa)$ l'ensemble des chemins [6] d'un ATPPD $\mathcal{A}$ muni de la valuation paramétrique κ. $Path_{\langle\bar{q},\mathbf{0}\rangle}(\mathcal{A}, \kappa)$ représente l'ensemble des chemins d'un ATPPD $\mathcal{A}$ ayant pour premier état $(\bar{q}, \mathbf{0})$ avec $\mathbf{0}$ la valuation pour laquelle toutes les horloges de $\mathcal{X}$ sont remises à zéro. Tout chemin d'un ATPPD $\mathcal{A}$ est décrit en fonction de la valuation paramétrique κ de la manière suivante :*

$$\omega = \langle q_0, \nu_0\rangle \xrightarrow{t_0}_\kappa \langle q_1, \nu_1\rangle \xrightarrow{t_1}_\kappa \cdots \xrightarrow{t_{n-1}}_\kappa \langle q_n, \nu_n\rangle.$$

Soit $\mathcal{A}$ un ATPPD. Soit $\Delta \subseteq \mathbb{N}^{\mathcal{P}}$ un ensemble de valuations paramétriques tel que $\mathcal{A}$ soit bien formé. On a vu que pour chaque ATPD $\mathcal{A}_\kappa$, tout état $q \in Q$ et toute valuation ν telle que $\langle q, \nu\rangle$ soit accessible dans $\mathcal{A}_\kappa$, toute transition de source q ne peut être tirée qu'au temps $Exit_\kappa(q, \nu)$ égal à $\kappa(\phi(q)) - \nu(\psi(q))$.

Proposition 2.5. *Dans un ATPPD $\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, \phi, \psi, inv, prob, \langle\tau_q\rangle_{q\in Q})$ muni d'une valuation paramétrique κ, pour tout état $q \in Q$ et pour toute valuation ν tels que $\langle q, \nu\rangle$ est accessible, il existe un unique t tel que $\langle q, \nu\rangle \xrightarrow{t}_\kappa \langle q', \nu'\rangle$ avec $t = Exit(q, \nu)$ et q' un état cible.*

En particulier, pour tout $\kappa \in \Delta$, tout chemin de l'ATPPD $\mathcal{A}$

$$\omega = \langle q_0, \nu_0\rangle \xrightarrow{t_0,e_0}_\kappa \langle q_1, \nu_1\rangle \xrightarrow{t_1,e_1}_\kappa \cdots \xrightarrow{t_{n-1},e_{n-1}}_\kappa \langle q_n, \nu_n\rangle,$$

t_{n-1} est uniquement déterminé par ν_0 et les transitions probabilistes e_j avec $j \in \{0, \cdots, n-2\}$.

[6] Dans un ATPPD, les termes *chemin* et *exécution* sont équivalents car le temps est fixé. Il existe une unique exécution sur un chemin donné.

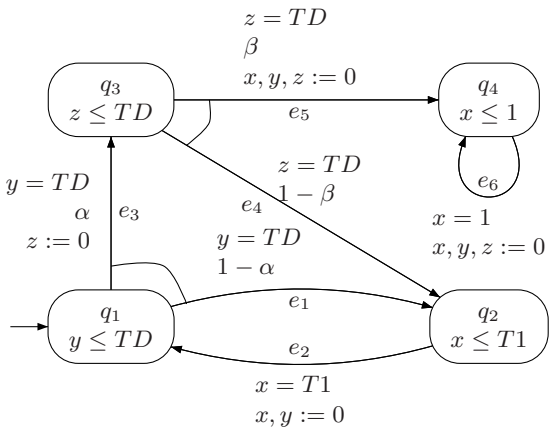

FIG. 2.12 – ATPPD du protocole BRP

Remarque 2.6. *A toute chemin ω de $\mathcal{A}$ correspond un chemin ω_κ dans $\mathcal{A}_\kappa$ pour toute valuation paramétrique $\kappa \in \Delta$. La probabilité de ω, $Pr(\omega)$, est égale à $Pr(\omega_\kappa)$. La durée de ω, $\kappa(Dur(\omega))$ notée $Dur_\kappa(\omega)$, est égale à $Dur(\omega_\kappa)$, soit $\sum_{k=0}^{n-1} Exit_\kappa(q_k, \nu_k)$.*

2.5.2.2 ATPPD associé à un ATPPSD

Un automate $\mathcal{A}$ est un ATPPD si $\mathcal{A}$ est un ATPPSD dont la famille de fonctions $\langle \tau_q \rangle_{q \in Q}$ associe pout tout état $q \in Q$ la garde $(\psi(q) = \phi(q))$.

Définition 2.23. *Soit $\mathcal{A}$ un ATPPD. Soit Δ un sous ensemble de $\mathbb{N}^{\mathcal{P}}$. Nous dirons que $\mathcal{A}$ est bien formé (resp. non zenon) pour Δ si pour toute valuation paramétrique $\kappa \in \Delta$, $\mathcal{A}_\kappa$ est un ATPD bien formé (resp. non zenon).*

Proposition 2.6. *Un ATPPSD $\mathcal{A}$ bien formé et fortement non zenon définit un ATPPD, noté ATPPD(A), bien formé et non zenon si la famille de fonctions $\langle \tau_q \rangle_{q \in Q}$ associe à toute distribution prob(q) une garde de la forme $\psi(q) = \phi(q)$.*

Démonstration: Comme pour les ATPSDs, nous associons à chaque ATPPSD un ATPPD noté $ATPPD(\mathcal{A})$, en remplaçant toute garde $0 \leq \psi(q) \leq \phi(q)$ par la garde $\psi(q) = \phi(q)$. Soit $\Delta \subseteq \mathbb{N}^{\mathcal{P}}$ un ensemble de valuations paramétriques pour lequel $\mathcal{A}$ est bien formé. D'après la définition 2.21, tout ATPSD $\mathcal{A}_\kappa$ pour $\kappa \in \Delta$ est bien formé. Ainsi, tout $ATPD(\mathcal{A}_\kappa)$ est bien formé et non zenon pour tout $\kappa \in \Delta$, d'après la proposition 2.3. Or, $(ATPD(\mathcal{A}_\kappa))_{\kappa \in \Delta} = (ATPPD(\mathcal{A}))_{\kappa \in \Delta}$. Alors $ATPPD(\mathcal{A})$ est bien formé et non zenon pour Δ d'après la définition 2.23. □

Exemple 2.10. *Prenons l'ATPPD $\mathcal{A}$ décrit par la figure 2.12. L'ATPD de la figure 2.4 de l'exemple 2.5 est équivalent à $\mathcal{A}$ muni de la valuation paramétrique κ avec $\kappa(T1) = 5$ et $\kappa(TD) = 2$. Le temps écoulé dans tout chemin $\omega \in Path(\mathcal{A}, \kappa)$ est déterminé par κ et la valuation initiale ν_0.*

Remarque 2.7. *Dans certains cas, on peut avoir un ATPPSD $\mathcal{A}$ zenon dont l'ATPPD associé est non zenon (voir l'étude de cas de la section 3.4). En effet, les gardes de la forme $\psi(q) \leq \phi(q)$ peuvent entraîner des exécutions de durée nulle si toute transition du cycle est tirée instantanément après l'entrée dans l'état.*

Soit un cycle de $\mathcal{A}$:

$$q_1 \stackrel{e_0}{\rightarrow} q_2 \stackrel{e_1}{\rightarrow} \cdots \stackrel{e_{n-1}}{\rightarrow} q_{n-1} \stackrel{e_n}{\rightarrow} q_1,$$

où $\psi(q_{n-1})$ est uniquement remise à zéro par e_0 et $\phi(q_{n-1}) > 0$. En considérant que $\mathcal{A}$ est bien formé, si la transition e_n peut être tirée à $\nu(\psi(q_{n-1})) = 0$, le cycle est de durée nulle.

Par ailleurs, les gardes de l'ATPPD associé à l'ATPPSD sont de la forme $\psi(q) = \phi(q)$ ce qui empêche ce type d'exécutions de se dérouler. Dans le cycle décrit, la transition e_n sera alors tirée à $\nu(\psi(q_{n-1})) = \phi(q_{n-1})$ dans $ATPPD(\mathcal{A})$.

2.5.2.3 Calcul du pire temps moyen de convergence dans un ATPPSD

Soit $\mathcal{A}$ un ATPPSD admettant un unique état final et absorbant q_{end}.

La proposition 2.4 montre que pour calculer le pire temps moyen de convergence (i.e temps moyen de convergence maximal) d'un ATPSD $\mathcal{A}$, on peut considérer l'ATPD correspondant $ATPD(\mathcal{A})$. Le calcul du pire temps moyen de convergence dans un ATPPSD se fait aussi en considérant l'ATPPD qui lui est associé. Cet ATPPD capture en effet le temps maximal écoulé en chaque état q de l'ATPPSD.

Proposition 2.7. *Etant donné un ATPPSD $\mathcal{A}$, le pire temps moyen de convergence est capturé par ATPPD($\mathcal{A}$).*

Démonstration: Soit Δ un ensemble de valuations paramétriques telles que l'ATPPSD $\mathcal{A}$ soit bien formé et fortement non zenon. Pour tout $\kappa \in \Delta$, l'ATPD $ATPD(\mathcal{A}_\kappa)$ capture le pire temps moyen de convergence de l'ATPSD $\mathcal{A}_\kappa$ d'après la proposition 2.4. Or, $(ATPD(\mathcal{A}_\kappa))_{\kappa\in\Delta} = ((ATPPD(\mathcal{A}))_{\kappa\in\Delta})$. Donc $ATTPD(\mathcal{A})$ capture le pire temps moyen de convergence de $\mathcal{A}$. □

Comme nous l'avons fait pour les ATPD dans la définition 2.19, nous définissons de même le temps moyen de convergence dans un ATPPD $\mathcal{A}$ muni d'une valuation paramétrique κ par :

$$ExpAbs(\mathcal{A}, \kappa) = \sum_{\omega\in\Omega_{q_{end}}} Pr(\omega)Dur_\kappa(\omega).$$

2.5.3 Motivation des ATPPD

Comme nous l'avons déjà fait remarqué, les ATPPDs correspondent à des chaînes de Markov à coûts positifs représentés par des paramètres. On s'intéresse aux ATPPDs car les chaînes de Markov sont plus faciles à manipuler que les processus de décision markovien [30] et ne contiennent pas de non déterminisme. Dans le cas d'évaluation de performance (par exemple, le calcul du pire temps moyen de convergence), des algorithmes efficaces peuvent ainsi être proposés sur les ATPPDs contrairement aux résultats EXPTIME-hard généralement obtenus dans les automates temporisés probabilistes [29] (voir l'exemple 2.11).

Par ailleurs, la discrétisation du temps a été la seule méthode proposée pour le calcul de temps moyen de convergence dans un ATPPD [24]. Elle se base sur la construction du système temporisé probabiliste $\mathcal{M}_\mathcal{A}$ associé à $\mathcal{A}$ (voir la section 1.5) mais en considérant cette fois-ci les valuations d'horloges ν dans $\mathbb{N}^\mathcal{X}$ et non pas dans $\mathbb{R}^\mathcal{X}$. Le temps s'écoule dans $\mathbb{N}$ au lieu de

$\mathbb{R}$ et donc toute transition dans $\mathcal{M}_{\mathcal{A}}$ qui représente l'écoulement du temps se fait à chaque unité de temps. Les gardes doivent aussi être fermées et ne sont pas des gardes diagonales.

Le calcul fait appel à des méthodes itératives pour la résolution de problèmes d'optimisation linéaire [7, 17, 16] sur les processus de décision markovien. On obtient au final deux bornes : un temps moyen maximal de convergence (i.e le pire temps moyen de convergence) et un temps moyen minimal de convergence vers un état spécifique.

Si nous voulons calculer le pire temps moyen de convergence vers un état final absorbant q_{end} d'un automate temporisé probabiliste paramétré $\mathcal{A}$ quelconque, nous pouvons dans certains cas obtenir un ATPPSD à partir de $\mathcal{A}$ en opérant certaines restrictions sur $\mathcal{A}$ dans le but de calculer le pire temps moyen de convergence vers un état final absorbant q_{end}.

Nous expliquons ci-dessous certaines des restrictions qui peuvent être imposées à $\mathcal{A}$:

1. Les contraintes temporelles peuvent être de forme assez simple pour nous permettre de déterminer manuellement l'adversaire qui reste le plus longtemps possible dans chaque état visité de $\mathcal{A}$ en vue de capturer le pire temps moyen de convergence (voir l'exemple 2.12).

2. De même, pour faire un choix entre les distributions possibles en chaque état de $\mathcal{A}$, on peut déterminer manuellement un adversaire qui opte pour la distribution qui capture le temps écoulé le plus long possible (voir l'exemple 2.12).

3. Si l'on impose des contraintes sur les paramètres qui apparaissent dans $\mathcal{A}$ pour assurer un comportement naturel du système modélisé (voir exemple 2.6), certaines formes de non déterminisme (notamment entre les distributions) peuvent disparaître.

4. dans le cas où $\mathcal{A}$ est le résultat d'une composition parallèle de plusieurs automates composants, à une seule horloge chacun par exemple (voir l'exemple 2.6 et la section 3.4), le non déterminisme peut être réduit en appliquant des réductions d'ordre partiel [4, 14]. Cette méthode nous permet dans un premier temps d'obtenir un modèle intermédiaire comprenant du non déterminisme, sur lequel nous pouvons ensuite appliquer certaines techniques manuelles pour en supprimer le non déterminisme.

Exemple 2.11. *Cet exemple permet de mieux comprendre le rôle des horloges et des fonctions qui leur sont associées ψ et ϕ dans l'expression des coûts dans un ATPPD. On montre aussi la différence entre l'approche suivie en utilisant les ATTPDs et celle donnée par la discrétisation du temps dans les automates temporisés probabilistes [24].*

Prenons l'ATPPD $\mathcal{A}$ décrit dans la figure 2.13. L'automate possède deux horloges x et y et k segments triangulaires tels que le ième segment soit de durée nulle ou égale à 2^i. Lorsque l'on entre dans l'état q, le temps écoulé depuis l'état initial où y est remis à zéro, est égal à un entier entre 0 et $2^k - 1$ (ou $\sum_{l=0}^{k-1} 2^l$). Ainsi, il est possible d'avoir en s un temps d'attente compris entre 1 et 2^k car $\psi(q) = y$ et $\phi(q) = 2^k$. Dans une chaîne de Markov obtenue à partir du graphe des zones de $\mathcal{A}$, comme cela est expliqué dans la sous section 2.2.2, et qui associe un coût unique à chaque état [22], nous devons distinguer chaque cas par un état différent.

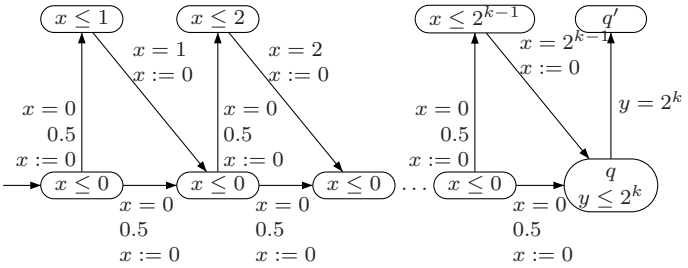

FIG. 2.13 – Automate de l'exemple 2.11

Ainsi, l'état q sera représenté dans cette chaîne de Markov par un nombre d'états exponentiel en k.

D'autre part, la discrétisation du temps dans les automates temporisés probabilistes se fait en construisant le système probabiliste temporisé $\mathcal{M}_{\mathcal{A}}$ associé à l'automate $\mathcal{A}$ où les états sont des états valués de la forme $\langle q, \nu \rangle$ (voir la section 1.5) où $\nu \in \mathbb{N}^{\mathcal{X}}$. Le nombre d'états générés en suivant cette approche est donc exponentiel en k car l'horloge y peut prendre des valeurs comprises entre 0 et 2^k.

La méthode que nous décrirons dans la section 3.2 permet de calculer en général le temps moyen de convergence vers un état final absorbant q_{end} (ici q') en évitant la construction de $\mathcal{M}_{\mathcal{A}}$ qui peut être de coût exponentiel.

Dans cet exemple simple, on constate que le temps moyen de convergence vers q' est de durée 2^k car tous les chemins acceptants sont de durée 2^k puisque l'horloge y n'est jamais remise à zéro.

Exemple 2.12. *On montre dans cet exemple comment déterminer manuellement un adversaire qui capture le pire temps dans l'automate temporisé probabiliste qui modélise le protocole IEEE 1394 (FireWire) root contention. On considère l'automate temporisé probabiliste I_1^p [26, 33] décrit dans la figure 2.14.*

Un choix probabiliste uniforme se fait au niveau de chaque noeud noir. Par exemple, de l'état fast_start on arrive à l'état fast_fast ou fast_slow avec une probabilité égale à 0.5.

Dans cet automate temporisé probabiliste, il existe deux sources de non déterminisme : l'une au niveau de la distribution à choisir dans les états start_start, fast_fast et slow_slow ; l'autre au niveau du temps d'attente dans chaque état, donné par son invariant et les gardes associées aux distributions de l'état en question.

Dans cet exemple, il est facile d'identifier un adversaire qui capture le temps maximal nécessaire pour atteindre l'état done. Le non déterminisme dans le choix des distributions se situe au niveau des états start_start, fast_fast et slow_slow. Dans les états fast_fast et slow_slow, l'adversaire doit choisir la distribution de fast_fast (resp. slow_slow) à start_start pour capturer le temps maximal. Dans l'état start_start, la distribution choisie importe peu car les états fast_start, start_fast, start_slow et slow_start ont la même garde $x \leq 360$. Par ailleurs, l'adversaire doit rester le plus longtemps possible dans chaque état, tout en vérifiant l'invariant et la garde de l'état en question. L'ATPD qui correspond à l'automate de la figure 2.14 est décrit dans la figure 2.15.

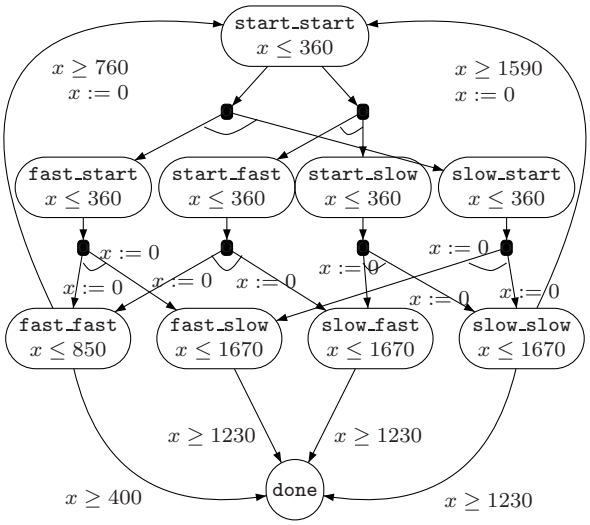

FIG. 2.14 – Automate temporisé probabiliste du protocole root contention

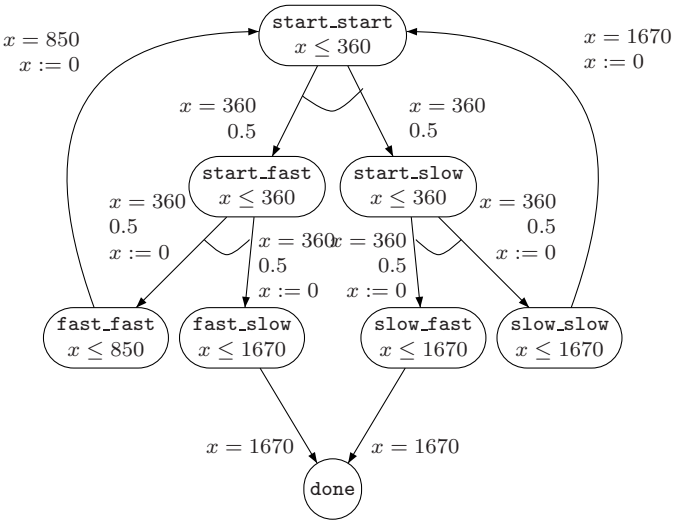

FIG. 2.15 – ATPD selon un adversaire du protocole root contention

Chapitre 3

Calcul du temps moyen de convergence sur les ATPPD

3.1 Transformation d'un ATPPD en une variante de chaîne de Markov avec coût : le graphe des macro-steps

Soit $\mathcal{A}$ un ATPPD admettant un unique état final absorbant q_{end}. Soit $\Delta \subseteq \mathbb{N}^{\mathcal{P}}$ un ensemble de valuations paramétriques telles que $\mathcal{A}$ soit bien formé. Cet automate paramétré représente un ensemble d'ATPDs $\{(\mathcal{A}_\kappa)_{\kappa\in\Delta}\}$ ayant un unique état final absorbant q_{end}.
Nous présentons une méthode de calcul des temps moyens de convergence vers q_{end} de chacun des automates $\mathcal{A}_\kappa$, qui repose sur un calcul paramétré fait directement sur l'ATPPD $\mathcal{A}$.
Dans ce but, nous construisons un automate à coûts, dont les coûts peuvent être éventuellement des paramètres, que nous appelons le *graphe des macro-steps*.

3.1.1 Macro-step

Soit $\mathcal{A} = (Q, \mathcal{P}, \bar{q}, \mathcal{X}, \phi, \psi, inv, prob, \langle\tau_q\rangle_{q\in Q})$ un ATPPD bien formé et non zenon. On note $\mathcal{E}_\mathcal{A}$ l'ensemble des transitions probabilistes de l'ATPPD $\mathcal{A}$.

Définition 3.1. *(Point.) On définit un* point *de $\mathcal{A}$ comme un couple (X, q) où X est une partie de l'ensemble des horloges de $\mathcal{A}$ ($X \subseteq \mathcal{X}$) et q un état de $\mathcal{A}$ ($q \in Q$).*

Remarque 3.1. *L'ensemble des points est un dépliage de l'ensemble des états de $\mathcal{A}$: le point (X, q) représente l'état q cible de transitions qui remettent à zéro exactement les horloges de l'ensemble X. Le point $(\mathcal{X}, \bar{q})$ est associé à l'état initial $\bar{q}$ car toutes les horloges sont remises à zéro à l'instant où l'on entre dans $\bar{q}$.*

Nous adoptons dans la suite la notation (q, X, q') au lieu de la notation $(q, \psi(q) = \phi(q), X, prob(q)(q'), q')$ pour décrire une transition probabiliste car dans un ATPPD la garde $(\psi(q) = \phi(q))$ est fonction de q et la probabilité $prob(q)(q')$ dépend de q et de q'.

Définition 3.2. *(Macro-step selon σ.) Soit (X, q) un point de $\mathcal{A}$.*
Soit $\sigma = e_1 e_2 \cdots e_m$ une suite finie de transitions probabilistes consécutives de $\mathcal{A}$ telle que $e_i = (q_{i-1}, Y_i, q_i)$ et qui satisfait les conditions suivantes :
- *$q_0 = q$,*
- *l'horloge $\psi(q_{m-1})$ appartient à X,*

– *l'horloge* $\psi(q_{m-1})$ *n'appartient pas à* Y_i *pour tout* $1 \leq i < m$.

Sous ces conditions, on dit que σ *définit un* macro-step *du point* (X, q) *au point* (Y_m, q_m). *On note ce macro-step* $(X, q) \stackrel{\sigma}{\Rightarrow} (Y_m, q_m)$.
A ce macro-step défini par σ, *nous associons les quantités suivantes :*
Le poids, *noté* $Wgt((X, q_0) \stackrel{\sigma}{\Rightarrow} (Y_m, q_m))$, *est égal à la probabilité du chemin défini par la séquence* σ. *Il est défini par :*

$$Wgt((X, q_0) \stackrel{\sigma}{\Rightarrow} (Y_m, q_m)) = \prod_{k=0}^{m-1} prob(q_k)(q_{k+1}).$$

La durée *de ce macro-step, notée* $Dur((X, q_0) \stackrel{\sigma}{\Rightarrow} (Y_m, q_m))$, *est définie par :*

$$Dur((X, q_0) \stackrel{\sigma}{\Rightarrow} (Y_m, q_m)) = \phi(q_{m-1}).$$

La longueur *est égale à* m.

Remarque 3.2. *Sous ces hypothèses, soit* ω *un chemin admissible de* $\mathcal{A}$

$$\omega = \langle q_0, \nu_0 \rangle \stackrel{e_0}{\rightarrow}_\kappa q_1 \stackrel{e_1}{\rightarrow}_\kappa q_2 \stackrel{e_2}{\rightarrow}_\kappa \cdots \stackrel{e_i}{\rightarrow}_\kappa q_i,$$

où e_i *est une transition qui remet* $\psi(q_{m-1})$ *à zéro. L'automate* $\mathcal{A}$ *étant bien formé et déterministe, le chemin* ω *peut être prolongé par le chemin* σ,

$$\sigma = q_i \stackrel{e_{i+1}}{\rightarrow}_\kappa \cdots q_{m-1} \stackrel{e_{m-1}}{\rightarrow}_\kappa q_m.$$

Comme $\psi(q_{m-1})$ *vaut zéro en entrant en* q_i, *n'est pas remise à zéro avant d'entrer en* q_m *et doit vérifier* $\psi(q_{m-1}) = \phi(q_{m-1})$ *sur la dernière transition, le temps écoulé entre l'entrée en* q_i *et l'entrée en* q_m *est exactement égal à* $\phi(q_{m-1})$. *On a aussi :*

$$Dur(\omega \cdot \sigma) = Dur(\omega) + Dur((Y_i, q_i) \stackrel{\sigma}{\Rightarrow} (Y_m, q_m)).$$

De plus, on a par définition :

$$Pr(\omega \cdot \sigma) = Pr(\omega) \cdot Wgt((Y_i, q_i) \stackrel{\sigma}{\Rightarrow} (Y_m, q_m)).$$

Exemple 3.1. *On considère l'ATPPD de l'exemple décrit dans la figure 2.12. On considère la séquence* σ_1 *composée de la suite de transitions probabilistes consécutives* e_1e_2. *L'horloge* x *n'est pas remise à zéro par la transition probabiliste* e_1. *L'horloge* x *est remise à zéro par la transition probabiliste* e_2 *entrant dans l'état source* q_1 *de* e_1 *et par la transition initiale qui remet toute les horloges à zéro. De plus,* $\psi(q_2) = x$ *et* $\phi(q_2) = T1$. *On en déduit deux macro-steps via* σ_1 *qui sont* $(\mathcal{X}, q_1) \stackrel{\sigma_1}{\Rightarrow} (\{x, y\}, q_1)$ *et* $(\{x, y\}, q_1) \stackrel{\sigma_1}{\Rightarrow} (\{x, y\}, q_1)$. *On a* $Dur((\mathcal{X}, q_1) \stackrel{\sigma_1}{\Rightarrow} (\{x, y\}, q_1)) = Dur((\{x, y\}, q_1) \stackrel{\sigma_1}{\Rightarrow} (\{x, y\}, q_1)) = T1$ *et* $Wgt((\mathcal{X}, q_1) \stackrel{\sigma_1}{\Rightarrow} (\{x, y\}, q_1)) = Wgt((\{x, y\}, q_1) \stackrel{\sigma_1}{\Rightarrow} (\{x, y\}, q_1)) = 1 - \alpha$.
D'autre part, on peut aussi déduire deux macro-steps via σ_2, *la suite de transitions probabilistes consécutives* $e_3e_4e_2$.
L'horloge x *est remise à zéro par la transition probabiliste* e_2 *entrant dans l'état source* q_1 *de* e_3 *et par la transition initiale qui remet toute les horloges à zéro. De plus,* $\psi(q_2) = x$ *et* $\phi(q_2) = T1$. *On en déduit deux macro-steps via* σ_2 *qui sont* $(\mathcal{X}, q_1) \stackrel{\sigma_2}{\Rightarrow} (\{x, y\}, q_1)$ *et* $(\{x, y\}, q_1) \stackrel{\sigma_2}{\Rightarrow} (\{x, y\}, q_1)$. *On a* $Dur((\mathcal{X}, q_1) \stackrel{\sigma_2}{\Rightarrow} (\{x, y\}, q_1)) = Dur((\{x, y\}, q_1) \stackrel{\sigma_2}{\Rightarrow} (\{x, y\}, q_1)) = T1$ *et* $Wgt((\mathcal{X}, q_1) \stackrel{\sigma_2}{\Rightarrow} (\{x, y\}, q_1)) = Wgt((\{x, y\}, q_1) \stackrel{\sigma_2}{\Rightarrow} (\{x, y\}, q_1)) = \alpha(1 - \beta)$.

Proposition 3.1. *Soit σ un chemin vérifiant les hypothèses de la définition 3.2 qui définit un macro-step selon σ. La longueur de ce macro-step est majorée par $|Q|$.*

Démonstration: On démontre que la suite de transitions probabilistes consécutives $\sigma = e_1e_2\cdots e_m = (q_0, Y_1, q_1)(q_1, Y_2, q_2)\cdots(q_{m-1}, Y_m, q_m)$ telle que q_0 est accessible et formant le macro-step $(X, q_0) \overset{\sigma}{\Rightarrow} (Y_m, q_m)$, ne contient pas deux fois le même état, soit $q_i \neq q_j\ \forall\ i, j \in \{0, \cdots, m\}$ et $i < j$. Par l'absurde :
Supposons qu'il existe $i < j$, $i, j \in \{0, \cdots, m\}$ tels que $q_i = q_j$.
Soit $\sigma' = (q_i, Y_i, q_{i+1})\cdots(q_{j-1}, Y_{j-1}, q_j) = e_ie_{i+1}\cdots e_{j-1}$ la sous séquence de σ entre ces deux états.
La durée du macro-step défini selon σ est fixée et est égale à $Dur((X, q_0) \overset{\sigma}{\Rightarrow} (Y_m, q_m))$. Par ailleurs, sous l'hypothèse d'ATPPD bien formé, on pourrait traverser σ' plus d'une fois car les états q_i et q_{j-1} ne sont pas bloquants. On pourrait donc avoir une séquence σ de la forme $(q_0, Y_1, q_1)(q_1, Y_2, q_2)\cdots(\sigma')^2\cdots(q_{m-1}, Y_m, q_m)$.
D'autre part, sous l'hypothèse d'ATPPD fortement non zenon, il existe dans le cycle σ' deux entiers $k, k' \in \{i, i+1, \cdots, j-1\}$ et une horloge $x \in \mathcal{X}$ tels que $x \in Y_k$, $\psi(q_{k'}) = x$ et $\phi(q_{k'}) \geq 1$. A chaque passage dans le cycle σ' une unité de temps est au moins écoulée. Ainsi, la durée de σ augmente strictement avec le nombre de passage dans σ' et ne peut rester égale à $Dur((X, q_0) \overset{\sigma}{\Rightarrow} (Y_m, q_m))$.
Il n'existe donc pas de cycle dans σ et le nombre d'états visités par σ est borné par le nombre d'états de l'ATPPD, soit $|Q|$. □

Remarque 3.3. *Soient (X, q) un point de $\mathcal{A}$ et $e = (t, Y, q')$ une transition probabiliste de cible q', telle que $\psi(t) \in X$. On considère les deux macro-steps $(X, q) \overset{\sigma_1}{\Rightarrow} (Y, q')$ et $(X, q) \overset{\sigma_2}{\Rightarrow} (Y, q')$, respectivement définis par deux suites σ_1 et σ_2 qui se terminent par la même transition probabiliste e. Alors $Dur((X, q) \overset{\sigma_1}{\Rightarrow} (Y, q')) = Dur((X, q) \overset{\sigma_2}{\Rightarrow} (Y, q'))$ mais les deux macro-steps n'ont pas nécessairement le même poids.*

La définition 3.3 permet de regrouper ces suites.

Notation 3.1. *Soient deux points (X, q) et (Y, q') et une transition probabiliste e de la forme (t, Y, q') et $\psi(t) \in X$. On note $EndSet((X, q), e, (Y, q'))$ l'ensemble des suites de transitions probabilistes σ ayant pour dernière transition e et définissant un macro-step de (X, q) à (Y, q').*

Remarque 3.4. *L'ensemble $EndSet((X, q), e, (Y, q'))$ est fini. En effet, grâce à la proposition 3.1, le nombre de macro-steps définis par des suites $\sigma \in EndSet((X, q), e, (Y, q'))$ est borné par $|\mathcal{T}|^{|Q|}$ où $\mathcal{T}$ correspond à l'ensemble des transitions de l'ATPPD $\mathcal{A}$.*

Définition 3.3. *(Macro-step selon la transition probabiliste e.) Soient (X, q) et (Y, q') deux points et soit e une transition probabiliste de la forme (t, Y, q') avec $\psi(t) \in X$. On suppose que $EndSet((X, q), e, (Y, q')) \neq \emptyset$. Sous ces conditions, on dit qu'il existe un macro-step de (X, q) à (Y, q') selon e, noté $(X, q) \overset{e}{\Rightarrow} (Y, q')$. A ce macro-step, nous associons les notions :*
Le poids *du macro-step de (X, q) à (Y, q') selon e, noté $Wgt((X, q) \overset{e}{\Rightarrow} (Y, q'))$ est égal à :*

$$\sum_{\sigma \in EndSet((X,q),e,(Y,q'))} Wgt((X, q) \overset{\sigma}{\Rightarrow} (Y, q')).$$

La durée *du macro-step de* (X,q) *à* (Y,q') *selon* e*, notée* $Dur((X,q) \overset{e}{\Rightarrow} (Y,q'))$ *est égale à la durée* $Dur((X,q) \overset{\sigma}{\Rightarrow} (Y,q'))$ *commune à tout macro-step défini par* σ *pour tout* $\sigma \in EndSet((X,q),e,(Y,q'))$.

Le poids du macro-step $(X,q) \overset{e}{\Rightarrow} (Y,q')$ représente la probabilité de l'ensemble des chemins qui partent de q et qui arrivent en q' par la transition probabiliste $e=(t,Y,q')$ sans remettre $\psi(t)$ à zéro. Ces chemins sont de durée égale $\phi(t)$.

Définition 3.4. *(Macro-step.) Soient* (X,q) *et* (Y,q') *deux points. Soit* $\mathcal{E}_{(Y,q')}$ *l'ensemble des transitions probabilistes de* $\mathcal{E}_{\mathcal{A}}$ *de la forme* (t,Y,q')*. On suppose qu'il existe* $e \in \mathcal{E}_{(Y,q')}$ *tel que* $EndSet((X,q),e,(Y,q')) \neq \emptyset$.

Sous ces conditions, on dit qu'il existe un macro-step de (X,q) *à* (Y,q') *qu'on note* $(X,q) \Rightarrow (Y,q')$.

A ce macro-step, sont associées les notions :

Le poids *du macro-step de* (X,q) *à* (Y,q')*, noté* $Wgt((X,q) \Rightarrow (Y,q'))$ *est égal à :*

$$\sum_{e \in \mathcal{E}_{(Y,q')}} Wgt((X,q) \Rightarrow (Y,q')).$$

La durée *du macro-step de* (X,q) *à* (Y,q')*, noté* $Dur((X,q) \Rightarrow (Y,q'))$ *est égal à :*

$$\sum_{e \in \mathcal{E}_{(Y,q')}} \frac{Wgt((X,q) \overset{e}{\Rightarrow} (Y,q')) \cdot Dur((X,q) \overset{e}{\Rightarrow} (Y,q'))}{Wgt((X,q) \Rightarrow (Y,q'))}.$$

Le poids du macro-step $(X,q) \Rightarrow (Y,q')$ représente la probabilité des chemins qui partent de q et qui arrivent en q' par les transitions de la forme $(-,Y,q')$. La durée ainsi définie est égale à la moyenne pondérée des durées de ces chemins.

3.1.2 Graphe des macro-steps

Dans la suite, on tiendra uniquement compte des points (X,q) de $\mathcal{A}$ tels qu'il existe une transition probabiliste de la forme $(-,X,q)$ appartenant à $\mathcal{E}_{\mathcal{A}}$.

Définition 3.5. *(Graphe des macro-steps.) On définit le graphe des macro-steps, noté* $MS(\mathcal{A})$*, comme étant le graphe dont les états sont les points et dont les transitions sont les macro-steps. On suppose qu'il existe un unique noeud de la forme* $(-,\bar{q})$ *(respectivement,* $(-,q_{end})$*), soit* $(\mathcal{X},\bar{q})$ *(respectivement,* $(\mathcal{X},q_{end})$*), qu'on note* $\bar{q}$ *(respectivment,* q_{end}*).*

Exemple 3.2. *On considère l'ATPPD décrit dans la figure 2.12. Reprenons les macro-steps* $(\mathcal{X},q_1) \overset{\sigma_1}{\Rightarrow} (\{x,y\},q_1)$ *et* $(\mathcal{X},q_1) \overset{\sigma_2}{\Rightarrow} (\{x,y\},q_1)$*, de* $(\mathcal{X},q_1)$ *à* $(\{x,y\},q_1)$ *(voir l'exemple 3.1). On a que* $\mathcal{E}_{(\{x,y\},q_1)} = e_2$ *et il n'existe aucun autre macro-step selon* σ *de* $(\mathcal{X},q_1)$ *à* $(\{x,y\},q_1)$*. On a* $EndSet((\mathcal{X},q_1),e_2,(\{x,y\},q_1)) = \{\sigma_1,\sigma_2\}$*. On peut ainsi déduire un macro-step* $(\mathcal{X},q_1) \Rightarrow (\{x,y\},q_1)$ *de poids* $Wgt((\mathcal{X},q_1) \Rightarrow (\{x,y\},q_1)) = Wgt((\mathcal{X},q_1) \overset{\sigma_1}{\Rightarrow} (\{x,y\},q_1)) + Wgt((\mathcal{X},q_1) \overset{\sigma_2}{\Rightarrow} (\{x,y\},q_1)) = (1-\alpha) + \alpha(1-\beta) = 1-\alpha\beta$ *et* $Dur((\mathcal{X},q_1) \Rightarrow (\{x,y\},q_1)) = \frac{(1-\alpha)\cdot T1 + \alpha(1-\beta)\cdot T1}{1-\alpha\beta} = T1$

3.1.3 Construction du graphe des macro-steps

Nous décrirons dans ce qui suit l'algorithme qui permet de construire le graphe de macro-steps $\widehat{MS}(\mathcal{A})$ associé à un ATPPD $\mathcal{A}$. Ce graphe est un sous graphe de $MS(\mathcal{A})$ car la construction de $\widehat{MS}(\mathcal{A})$ ne tient pas compte de tous les points de $MS(\mathcal{A})$.

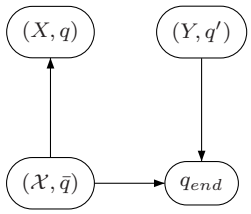

FIG. 3.1 – Graphe des macro-steps $MS(\mathcal{A})$ de l'exemple 3.3

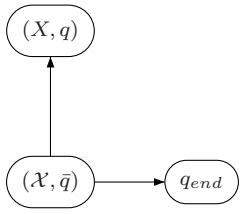

FIG. 3.2 – Le graphe des macro-steps $\widehat{MS}(\mathcal{A})$ obtenu par une démarche en avant de l'exemple 3.3

3.1.3.1 Description générale

Pour obtenir le graphe des macro-steps à partir de $\mathcal{A}$, deux techniques peuvent être utilisées :

1. une démarche en avant qui permet de calculer l'ensemble des successeurs de la forme (Y, q') de tout point (X, q) tel que $(X, q) \Rightarrow (Y, q')$ est un macro-step. Elle va permettre de construire tous les points accessibles à partir de $(\mathcal{X}, \bar{q})$.
2. une démarche en arrière qui permet de calculer l'ensemble des prédécesseurs (Y, q') de tout point (X, q) tel que $(Y, q') \Rightarrow (X, q)$ est un macro-step. Elle permet de construire tous les points qui permettent d'atteindre $(\mathcal{X}, q_{end})$.

Par la suite, nous aurons besoin pour le calcul du graphe des macro-steps $\widehat{MS}(\mathcal{A})$ des points à la fois accessibles de $(\mathcal{X}, \bar{q})$ et qui atteignent q_{end}. Le graphe $\widehat{MS}(\mathcal{A})$ représente donc l'ensemble $Acc(MS(\mathcal{A})) \cap coAcc(MS(\mathcal{A}))$ où $Acc(MS(\mathcal{A}))$ est l'ensemble des points accessibles de $(\mathcal{X}, \bar{q})$ et $coAcc(MS(\mathcal{A}))$ l'ensemble des points qui permettent d'atteindre $(\mathcal{X}, q_{end})$. Ceci peut être fait grâce aux algorithmes usuels d'acessibilité dans les graphes [13, 35].

Exemple 3.3. *On considère un graphe des macro-steps $MS(\mathcal{A})$ décrit dans la figure 3.1. Si l'on adopte une démarche en avant sur $MS(\mathcal{A})$, on obtient le graphe des macro-steps de $\mathcal{A}$ $\widehat{MS}(\mathcal{A})$ décrit dans la figure 3.2. D'autre part, si l'on adopte une démarche en arrière sur $MS(\mathcal{A})$, on obtient le graphe des macro-steps de $\mathcal{A}$ $\widehat{MS}(\mathcal{A})$ décrit dans figure 3.3. Enfin, le graphe des macro-steps $\widehat{MS}(\mathcal{A})$ dont nous aurons besoin par la suite est décrit dans la figure 3.4. Il est le résultat de l'intersection des deux graphes obtenus respectivement en avant et en arrière.*

Nous présentons ici la démarche en arrière sur l'ATPPD $\mathcal{A}$.

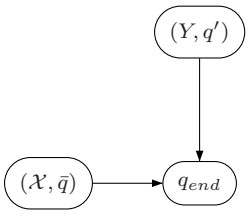

FIG. 3.3 – Le graphe des macro-steps $\widehat{MS}(\mathcal{A})$ obtenu par une démarche en arrière de l'exemple 3.3

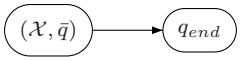

FIG. 3.4 – Le graphe des macro-steps $\widehat{MS}(\mathcal{A})$ final de l'exemple 3.3

Définition 3.6. *Etant donné un point* (X, q)*, on définit l'*ensemble des prédecesseurs $Pre(X, q)$ *de* (X, q) *comme étant l'ensemble* $\{(Y, q') \mid (Y, q') \Rightarrow (X, q)\}$ *dans* $MS(\mathcal{A})$.

On peut ainsi définir la fonction Pre (respectivment, $Post$) qui, pour tout point (X, q), permet de calculer l'ensemble des prédecesseurs (respectivment, successeurs) de (X, q), $Pre(X, q)$ (respectivement, $Post(X, q)$).

Si l'on applique la fonction Pre (respectivement, $Post$) à un ensemble de points V, on note $Pre(V) = \bigcup_{(X,q)\in V} Pre(X, q)$.

On fait appel itérativement à la fonction Pre décrite par l'algorithme 7 de la sous section 3.1.3.2 car nous adoptons une démarche en arrière. Pour tout point (X, q) obtenu par l'algorithme, on calcule $Pre(X, q)$ jusqu'à ne plus avoir de nouveau points à traiter. Pour toute transition probabiliste $(q', X, q) \in \mathcal{E}_{(X,q)}$ on considère l'horloge $\psi(q')$. On parcourt en arrière les chemins de l'ATPPD $\mathcal{A}$ en partant de q jusqu'à trouver la première transition probabiliste $e = (-, Y, q')$ qui remet $\psi(q')$ à zéro. On crée ainsi le point (Y, q') et le macro-step $(Y, q') \Rightarrow (X, q)$. Le point (Y, q') est ajouté à l'ensemble $Pre(X, q)$.

On commence par appliquer la fonction Pre à l'unique état final absorbant q_{end}. On calcule ainsi successivement $E_0 = Pre(q_{end})$, $E_1 = Pre(E_0)$, $\cdots \mathrm{E}_k = Pre(E_{k-1})$, $\cdots$ avec E_i un ensemble de points de $\mathcal{A}$. On calcule ainsi itérativement $Pre^*(q_{end}) = \cup_{i\in\mathbb{N}} Pre^i(q_{end})$. Le nombre de points d'un ATPPD étant borné [1] par $|\mathcal{E}_\mathcal{A}| + 1$, il existe un entier $k \in \mathbb{N}$ tel que $k \leq |\mathcal{E}_\mathcal{A}| + 1$ et $\cup_{i\in\mathbb{N}} Pre^i(q_{end}) = \cup_{i=1}^{k} Pre^i(q_{end})$. Nous considérons finalement l'ensemble $\mathcal{V} = Pre^*(q_{end})$.

3.1.3.2 Pseudo-algorithme de construction du graphe des macro-steps

Nous décrivons dans ce qui suit quatre pseudo-algorithmes dont trois fonctions, $\mathcal{E}$, $Pred_{directs}$ et Pre, et la fonction principale de calcul du graphe des macro-steps. Le poids des macro-

[1] On tient uniquement compte des points (X, q) de $\mathcal{A}$ tels qu'il existe une transition probabiliste de la forme $(-, X, q)$ appartenant à $\mathcal{E}_\mathcal{A}$. Le nombre de points d'un ATPPD est donc inférieur à $|Q| \times 2^{|\mathcal{X}|}$.

steps est calculé séparément par des algorithmes standards d'accessibilité dans les chaînes de Markov [12].

Pour un point (X, q) de $\mathcal{A}$, la fonction $\mathcal{E}$ permet de calculer les transitions de $\mathcal{E}_\mathcal{A}$ qui ont pour état cible q et qui remettent l'ensemble d'horloges X à zéro. Ces transitions sont de la forme $(-, X, q)$. On obtient ainsi l'ensemble $\mathcal{E}_{(X,q)}$.

Algorithme 5 Calcul de $\mathcal{E}_{(X,q)}$ par la fonction $\mathcal{E}(X, q)$.

for all $e \in \mathcal{E}_\mathcal{A}$ **do**
  if $cible(e) = q$ & $reset(e) = X$ **then**
    $\mathcal{E}_{(X,q)} := \mathcal{E}_{(X,q)} \cup e$
  end if
end for
return $\mathcal{E}_{(X,q)}$;

Pour une transition ed de $\mathcal{A}$, la fonction $Pred_{directs}$ permet de calculer l'ensemble des transitions $e \in \mathcal{E}_\mathcal{A}$ telles que $cible(e) = source(ed)$.

Algorithme 6 Prédecesseurs directs d'une transition probabiliste ed : $Pred_{directs}(ed)$

$Pred := \emptyset$;
for all $e \in \mathcal{E}_\mathcal{A}$ **do**
  if $cible(e) = source(ed)$ **then**
    $Pred := Pred \cup e$;
  end if
end for
return $Pred$;

La fonction Pre de l'algorithme 7 prend en paramètres une transition ed, une horloge x, un paramètre d, un ensemble A de transitions et un ensemble MS de macro-steps, tous deux initialement vides.

Elle retourne la liste des uplets $((Y, t) \Rightarrow (X, q), d)$ tels que :
- $ed = (q', X, q)$,
- t est la cible d'une transition e telle que $reset(e) = Y$ avec $x \in Y$,
- il existe au moins un chemin de source t et de cible q sur lequel x n'est jamais remise à zéro et se terminant par ed. Ce chemin est de durée d.

Lorsque l'on appelle la fonction Pre sur la transition $ed = (q', X, q)$ avec l'horloge $x = \psi(q')$, le paramètre $d = \phi(q')$ alors $Pre(ed, \psi(q'), \phi(q'), \emptyset, \emptyset)$ fournit la liste des macro-steps de cible (X, q). La fonction Dur leur assigne leur durée $\phi(q')$. Le calcul du poids des macro-steps de la forme $(Y, t) \Rightarrow (X, q)$ se fait séparément en utilisant les algorithmes standards d'accessibilité dans les chaînes de Markov comme cela est décrit dans les algorithmes 8 et 9.

Nous décrivons maintenant l'algorithme 7 de la fonction Pre :
On cherche itérativement par un parcours en arrière de $\mathcal{A}$, les transitions qui remettent x à zéro. On fait ainsi appel récursivement à la fonction Pre sur chaque transition calculée par $Pred_{directs}$ et qui ne remet pas x à zéro. On fait appel à la fonction Pre au plus $|\mathcal{E}_\mathcal{A}|$ fois.

L'ensemble $Pred$ représente les prédécesseurs directs de la transition en argument de la fonction Pre à chaque appel de cette dernière. Ainsi, pour toute transition e calculée par la fonction $Pred_{directs}$, si e remet l'horloge x à zéro on arrête l'itération (i.e parcours en arrière) sur cette transition et on crée le macro-step correspondant sinon, on appelle de nouveau la fonction Pre sur e si cet appel n'a pas été fait précédemment dans le calcul. Les transitions sur lesquelles la fontion Pre a déjà été appelée sont stockées dans l'ensemble A.

Pour $e \in Pred$, remettant l'horloge x à zéro et tel que pour tout $e' \in E$, $reset(e) \neq reset(e')$, on crée le macro-step ms égal à $(reset(e), cible(e)) \Rightarrow (X, q)$ et on rajoute la transition e à l'ensemble E. L'ensemble MS permet de stocker les macro-steps créés selon ed et de durée d.

On ne calcule pas l'ensemble des chemins $EndSet((Y,t), e, (X,q))$ et on forme uniquement un macro-step $(Y,t) \Rightarrow (X,q)$ selon e sans se préoccuper du calcul de son poids dans cet algorithme. On parcourt en effet au moins un chemin de l'ensemble $EndSet((Y,t), e, (X,q))$ pour créer le macro-step associé. Le poids Wgt associé au macro-step sera calculé grâce aux algorithmes 8 et 9.

Algorithme 7 Calcul de $Pre(X,q)$ suivant la transition $ed = (q', X, q)$: la fonction $Pre(ed, x, d, A, MS)$

$Pred := Pred_{directs}(ed)$;
$A := A \cup ed$;
$E := \emptyset$;
for all $e \in Pred$ **do**
  if $source(e) = \bar{q}$ & $(\mathcal{X}, \bar{q}) \Rightarrow (X,q) \notin MS$ **then**
    $MS := MS \cup ((\mathcal{X}, \bar{q}) \Rightarrow (X,q))$
  end if
  if $x \in reset(e)$ **then**
    if $(\forall e' \in E \mid reset(e) \neq reset(e'))$ **then**
      $E := E \cup \{e\}$;
      $Z := reset(e)$;
      $l := cible(e)$;
      $ms := (Z,l) \Rightarrow (X,q)$;
      $Dur(ms) = d$;
      if $ms \notin MS$ **then**
        $MS := MS \cup (ms, Dur(ms))$;
      end if
    else
      continue;
    end if
  else
    if $\forall t \in A \mid (t \neq e)$ **then**
      $MS := MS \cup Pre(e, x, d, A, MS)$;
    else
      continue;
    end if
  end if
end for
return MS ;

Le calcul des poids des macro-steps de l'ensemble MS fait appel aux algorithmes standards [12] d'accessibilité sur les chaînes de Markov. Notre calcul repose sur l'ATPPD $\mathcal{A}$ qui est une chaîne de Markov à coûts. Pour un macro-step $(Y,t) \Rightarrow (X,q)$ de l'ensemble MS, on doit calculer la probabilité de l'ensemble des chemins partant de t, de longueur inférieure ou égale à $|Q|$, se terminant par ed et ne remettant jamais l'horloge x à zéro.
Nous devons tout d'abord calculer la probabilité de l'ensemble des chemins partant de t, de longueur i avec $i \leq |Q|$, se terminant par ed et ne remettant jamais l'horloge x à zéro.
Ce calcul fait appel à une matrice M_{ed} de taille $(|Q|, |Q|)$ telle que $M_{ed} = (m_{i,l})_{\substack{i \in \{1,\cdots,|Q|\} \\ l \in Q}}$ où $m_{i,l}$ représente la probabilité de l'ensemble des chemins partant de l, de longueur i se terminant par ed et ne remettant jamais l'horloge x à zéro.

L'algorithme 8 permet de calculer la matrice M_{ed}. Pour chaque état $l \in Q$, on introduit l'ensemble $Post(l,x)$ qui regroupe les transitions de source l et qui ne remettent pas l'horloge x à zéro.

On calcule ainsi itérativement $m_{i,l}$ pour tout $i \in \{1, \cdots, |Q|\}$ en initialisant $m_{1,l}$ à $prob(source(ed))(cible(ed))$ car les chemins se terminent par ed. Tout $m_{i,l}$ avec $i \geq 2$, est la somme des probabilités des chemins de longueur i, partant de l et dont la première transition doit appartenir à $Post(l, x)$ car l'horloge x ne doit pas être remise à zéro avant d'atteindre ed. La probabilité d'un de ces chemins est donc égale au produit de la probabilité d'une transition $e \in Post(l, x)$ par la probabilité du reste du chemin qui est de longueur $i - 1$ et partant de $cible(e)$.

Algorithme 8 Calcul de la probabilité des chemins de $l \in Q$ à $q = cible(ed)$ de longueur $\leq |Q|$, se terminant par $ed = (q', X, q)$ et ne remettant jamais x à zéro.

for all $l \in Q$ **do**
  if $l = q'$ **then**
    $m_{1,l} := prob(source(ed))(cible(ed))$;
  else
    $m_{1,l} := 0$;
  end if
end for
for $i = 2$ to $|Q|$ **do**
  for all $l \in Q$ **do**
    $m_{i,l} := 0$;
    for all $e \in Post(l, x)$ **do**
      $m_{i,l} := m_{i,l} + m_{i-1}, cible(e) \star prob(source(e))(cible(e))$;
    end for
  end for
end for
return $M_{ed} = (m_{i,l})_{\substack{i \in \{1,\cdots,|Q|\} \\ l \in Q}}$

Après avoir calculé la probabilité des chemins de longueur i, partant de l, se terminant par ed et ne remettant jamais x à zéro, pour tout $i \in \{1, \cdots, |Q|\}$ et tout $l \in |Q|$, on peut déduire le poids de chaque macro-step selon ed de l'ensemble MS calculé par l'algorithme 7. Pour tout $ms = (Y, t) \Rightarrow (X, q) \in MS$, $Wgt(ms)$ est la probabilité de l'ensemble des chemins de longueur inférieure ou égale à $|Q|$, partant de t, se terminant par ed et ne remettant jamais l'horloge x à zéro. Cette probabilité se calcule grâce à la matrice M_{ed} car elle est égale à la somme des probabilités des chemins de longueur i inférieure ou égale à $|Q|$, partant de t, se terminant par ed et ne remettant jamais x à zéro.

Algorithme 9 Calcul des poids des macro-steps selon ed

for all $ms = ((Y, t) \Rightarrow (X, q)) \in MS$ **do**
  return $Wgt(ms) = \sum_{i=1}^{|Q|} m_{i,t}$;
end for

L'algorithme 10 décrit la fonction principale de calcul du graphe des macro-steps. L'ensemble $Points$ représente les points de la forme (X, q) qui doivent être traités. Au cours du calcul, de nouveaux points sont créés puis rajoutés à l'ensemble $Points$ pour être ultérieurement

traités. Le calcul repose essentiellement sur l'appel de la fonction Pre pour tout point dans l'ensemble $Points$.

Pour tout $(X, q) \in Points$, on calcule l'ensemble des transitions entrant dans le point (X, q) par la fonction $\mathcal{E}$. On obtient $\mathcal{E}_{(X,q)}$. Cet ensemble est représenté par $Transitions_{entrantes}$. Pour toute transition $e \in Transitions_{entrantes}$, on fait appel à la fonction Pre avec en arguments e, $\psi(source(e))$, $prob(source(e))(cible(e))$, $\phi(source(e))$ et $\emptyset$ car nous voulons construire les macro-steps suivant e de $\mathcal{A}$.

L'ensemble MS regroupe les macro-steps créés par Pre pour chaque $e \in Transitions_{entrantes}$ et ayant tous (X, q) pour point cible. Ils sont de la forme $(Y, t) \Rightarrow (X, q)$. On fusionne ensuite les macro-steps de MS de même point source (Y, t). Ainsi si $ms, ms' \in MS$ ont le même point source (Y, t), on ne garde qu'un seul macro-step. Il aura pour poids la somme des poids de ms et ms' et pour durée la somme pondérée des durées de ms et ms'. Pour tout macro-step $ms \in MS$ de la forme $(Y, t) \Rightarrow (X, q)$, on ajoute le point (Y, t) à l'ensemble $Points$ si $(Y, t) \notin Points$ et qu'il n'a pas été déjà traité. L'ensemble $Points_{traites}$ regroupe les points déjà traités et qu'on ne doit plus visiter de nouveau. A ce niveau, l'ensemble MS représente l'ensemble $Pre(X, q)$ qui est ainsi rajouté à l'ensemble des macro-steps qui constitueront le graphe final des macro-steps $\widehat{MS}$.

Le point dont les points prédécesseurs ont déjà été calculés est supprimé de l'ensemble $Points$ et rajouté à l'ensemble $Points_{traites}$.
Dans la section 3.4, on détaillera la construction d'un tel graphe sur un exemple qui repose sur la modélisation du protocole de communication CSMA/CD [27, 28, 1].

Algorithme 10 Graphe des macro-steps

$Points = \{(\mathcal{X}, q_{end})\}$;
$Points_{traites} = \emptyset$;
$MS := \emptyset$;
$Transitions_{entrantes} := \emptyset$;
$\widehat{MS} = \emptyset$;
repeat
  Take and Remove (X, q) from $Points$.
  $Transitions_{entrantes} := \mathcal{E}(X, q)$;
  for all $e \in Transitions_{entrantes}$ **do**
    $MS := MS \cup Pre(e, \psi(source(e)), prob((source(e))(cible(e)), \emptyset, \emptyset)$;
  end for
  for all $ms = ((Y, t) \Rightarrow (X, q) \in MS$ **do**
    for all $ms' = ((Y', t') \Rightarrow (X, q) \in MS$ **do**
      if $(Y = Y')$ & $(t = t')$ **then**
        $Wgt(ms) := Wgt(ms) + Wgt(ms')$;
        $Dur(ms) := \frac{Dur(ms) \cdot Wgt(ms) + Dur(ms') \cdot Wgt(ms')}{Wgt(ms) + Wgt(ms')}$;
        $MS := MS \setminus ms'$;
      end if
    end for
  end for
    // L'ensemble MS représente à ce niveau $Pre(X, q)$.
  $\widehat{MS} := \widehat{MS} \cup MS$;
  for all $ms = ((Y, t) \Rightarrow (X, q)) \in MS$ **do**
    if $((Y, t) \notin Points)$ & $(Y, t) \neq (\mathcal{X}, \bar{q})$ & $(Y, t) \notin Points_{traites}$ **then**
      $Points := Points \cup (Y, t)$;
    end if
  end for
  $Points_{traites} := Points_{traites} \cup (X, q)$;
  $MS := \emptyset$;
  $Transitions_{entrantes} := \emptyset$;
until $Points = \emptyset$;
return $\widehat{MS}$;

Exemple 3.4. *Prenons l'ATPPD de BRP de l'exemple 3.1 et soit $q_4 = q_{end}$. Le graphe des macro-steps est décrit dans la figure 3.5. On calcule itérativement les prédécesseurs de q_{end}, $Pre^*(q_{end})$.*

L'ensemble $\widehat{MS}$ est initialement vide.

L'ensemble Points est initialisé à $(\mathcal{X}, q_4)$. On calcule l'ensemble des transitions de cible $(\mathcal{X}, q_4)$, soit l'ensemble $\mathcal{E}_{(\mathcal{X}, q_4)} = \{e_5\}$. On fait ensuite appel à la fonction Pre de l'algorithme 7 sur la transition e_5. On a $Pred_{directs}(e_5) = \{e_3\}$, $z = \psi(q_3)$ et $z \in reset(e_3)$. On crée ainsi le macro-step

$$(\{z\}, q_3) \Rightarrow q_{end}$$

de durée $Dur((\{z\}, q_3) \Rightarrow q_{end})$ égale à TD car $\phi(q_3) = TD$. Le poids de ce macro-step est égal à la probabilité de l'unique chemin de q_3 à q_4 qui est la transition e_5.

Donc $Wgt((\{z\}, q_3) \Rightarrow q_{end}) = \beta$.

On rajoute le point $(\mathcal{X}, q_4)$ *à l'ensemble* $Points_{traites}$ *initialement vide et le point* $(\{z\}, q_3)$ *à l'ensemble* $Points$.

On calcule l'ensemble des transitions de cible $(\{z\}, q_3)$, *soit l'ensemble* $\mathcal{E}_{(\{z\},q_3)} = \{e_3\}$. *En appliquant la fonction Pre, on a* $Pred_{directs}(e_3) = \{e_2\}$, $y = \psi(q_1)$ *et* $y \in reset(e_2)$. *De plus, on constate que* $source(e_3) = q_1$ *qui est l'état initial où toutes les horloges sont remises à zéro. On crée ainsi deux macro-steps selon* e_3,

$$(\{x, y\}, q_1) \Rightarrow (\{z\}, q_3)$$

et

$$(\mathcal{X}, q_1) \Rightarrow (\{z\}, q_3)$$

de durée TD. *Le poids de chacun de ces macro-steps est égale à* α *qui est la probabilité de l'unique chemin de* q_1 *à* q_3 *qui est la transition* e_3. *Donc* $Wgt((\{x, y\}, q_1) \Rightarrow (\{z\}, q_3)) = Wgt((\mathcal{X}, q_1) \Rightarrow (\{z\}, q_3)) = \alpha$.

On rajoute le point $(\{z\}, q_3)$ *à l'ensemble* $Points_{traites}$ *et le point* $(\{x, y\}, q_1)$ *à l'ensemble* $Points$.

L'ensemble des transitions $\mathcal{E}_{(\{x,y\},q_1)}$ *de cible* $(\{x, y\}, q_1)$ *est égal à* e_2.
En appliquant la fonction Pre, on a $Pred_{directs}(e_2) = \{e_1, e_4\}$ *et* $x = \psi(q_2)$. *Aucune des ces deux transitions ne remet l'horloge* x *à zéro. On appelle donc de nouveau la fonction Pre respectivement sur les transitions* e_1 *et* e_4. *L'ensemble A est à ce niveau égal à* $\{e_1, e_2, e_4\}$. *On a* $Pred_{directs}(e_1) = \{e_2\}$ *et* $Pred_{directs}(e_4) = \{e_3\}$. *La transition* e_2 *remet l'horloge* x *à zéro et* $source(e_1) = q_1$ *(l'état initial). On crée ainsi deux macro-steps* $(\{x, y\}, q_1) \Rightarrow (\{x, y\}, q_1)$ *et* $(\mathcal{X}, q_1) \Rightarrow (\{x, y\}, q_1)$ *de durée* $T1$.
Ainsi, $MS = \{((\{x, y\}, q_1) \Rightarrow (\{x, y\}, q_1), T1), ((\mathcal{X}, q_1) \Rightarrow (\{x, y\}, q_1), T1)\}$.

D'un autre côté, e_3 *ne remet pas* x *à zéro. On appelle de nouveau la fonction Pre sur* e_3 *et A devient égal à* $\{e_1, e_2, e_3, e_4\}$. *On a* $Pred_{directs}(e_3) = \{e_2\}$. *La transition* e_2 *remet l'horloge* x *à zéro et* $source(e_3) = q_1$. *On obtient deux macro-steps* $(\{x, y\}, q_1) \Rightarrow (\{x, y\}, q_1)$ *et* $(\mathcal{X}, q_1) \Rightarrow (\{x, y\}, q_1)$ *de durée* $T1$ *chacun mais qui appartiennent déjà à* MS. *Ils ne sont donc pas rajoutés à l'ensemble* MS.
Par ailleurs, le poids de chacun de ces macro-steps est calculé séparément.

Le poids du macro-step $(\{x, y\}, q_1) \Rightarrow (\{x, y\}, q_1)$ *est égal à la probabilité de l'ensemble des chemins partant de* q_1, *se terminant par* e_2 *et ne remettant pas* x *à zéro. On trouve les deux chemins suivants :*

$$q_1 \xrightarrow{e_1} q_2 \xrightarrow{e_2} q_1$$

qui est de probabilité $m_{2,q_1} = 1 - \alpha$ *et*

$$q_1 \xrightarrow{e_3} q_3 \xrightarrow{e_4} q_2 \xrightarrow{e_2} q_1$$

qui est de probabilité $m_{3,q_1} = \alpha(1 - \beta)$.

Ainsi $Wgt((\{x, y\}, q_1) \Rightarrow (\{x, y\}, q_1))$ *est égal à la somme des probabilités de ces deux chemins soit :*

$$(1 - \alpha) + \alpha(1 - \beta) = 1 - \alpha\beta.$$

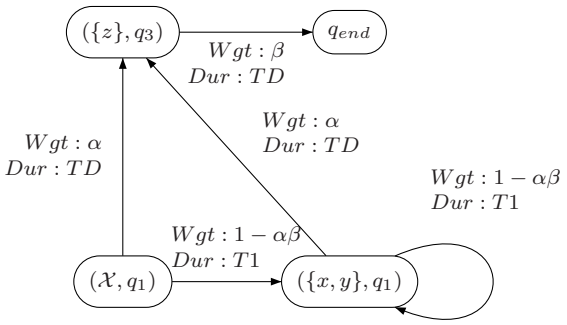

FIG. 3.5 – Graphe des macro-steps du protocole BRP

De même, le poids du macro-step $(\mathcal{X}, q_1) \Rightarrow (\{x, y\}, q_1)$ *est égal à la probabilité de l'ensemble des chemins partant de* q_1*, se terminant par* e_2 *et ne remettant pas* x *à zéro. Le poids de* $(\mathcal{X}, q_1) \Rightarrow (\{x, y\}, q_1)$ *est donc égal au poids du macro-step* $(\{x, y\}, q_1) \Rightarrow (\{x, y\}, q_1)$ *qui lui est égal à* $1 - \alpha\beta$*.*

A ce stade l'ensemble Points est vide, on arrête donc le parcours de l'ATPPD.
Ainsi, le calcul de $Pre^*(q_{end})$ *se termine et on a* $\mathcal{V} = Pre^*(q_{end})$*.*

Enfin, la matrice de transitions associée au graphe des macro-steps définie sur $\mathcal{V} = \{(\mathcal{X}, q_1), (\{x, y\}, q_1), (\{z\}, q_3), q_{end}\}$ *est la suivante :*

$$\begin{pmatrix} 0 & 1-\alpha\beta & \alpha & 0 \\ 0 & 1-\alpha\beta & \alpha & 0 \\ 0 & 0 & 0 & \beta \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

3.1.4 Complexité du graphe des macro-steps

Soit $\mathcal{A}$ un ATPPD et $\widehat{MS}(\mathcal{A})$ le graphe des macro-steps associé à $\mathcal{A}$.

Nous montrons que la construction du graphe des macro-steps $\widehat{MS}(\mathcal{A})$ est polynomial en $|\mathcal{E}_\mathcal{A}|$ qui est le nombre de transitions probabilistes de l'ATPPD $\mathcal{A}$.

Afin de générer la relation $\Rightarrow$ entre les points de $\mathcal{A}$, le calcul des ensembles $EndSet()$ (voir la notation 3.1 et la remarque 3.4) peut être de taille exponentielle en $|\mathcal{E}_\mathcal{A}|$. En effet, pour calculer l'ensemble $EndSet((X, q), e, (Y, q'))$, il faudrait calculer l'ensemble des chemins $\{\mathcal{E}_\mathcal{A}{}^* \cdot e\}$ entre (X, q) et (Y, q'), se terminant par e et ne remettant pas l'horloge $\psi(source(e))$ à zéro. Ces chemins ne contiennent pas de cycles d'après la proposition 3.1. Le nombre de ces chemins peut être exponentiel, égal à $|\mathcal{E}_\mathcal{A}|!$. Si un tel calcul veut être fait, nous pouvons calculer le poids du macro-step selon e directement par l'algorithme 7 en ajoutant une valeur flottante dans ses paramètres et en appelant Pre éventuellement un nombre exponentiel de fois.
Notre technique ne demande cependant pas le calcul des ensembles $EndSet()$. En particulier, pour tout couple de points $((X, q), (Y, q'))$ de $\mathcal{A}$ et toute transition probabiliste $e =$

$(q'', Y, q') \in \mathcal{E}_{\mathcal{A}}$, le poids $Wgt((X,q) \stackrel{e}{\Rightarrow} (Y,q'))$ de la définition 3.3 est calculé grâce aux algorithmes probabilistes standards d'accessibilité appliqués à $Untimed(\mathcal{A})$ [25] sans avoir besoin de calculer l'ensemble $EndSet((X,q),e,(Y,q'))$.

En effet, l'algorithme 7 calcule l'ensemble des macro-steps selon e sans calculer le poids associé à chacun et renvoie l'ensemble MS des macro-steps selon e, tous de même durée $\phi(source(e))$. Il est polynomial en $O(|\mathcal{E}_{\mathcal{A}}|^2)$ car on appelle au plus $|\mathcal{E}_{\mathcal{A}}|$ fois la fonction Pre.

D'autre part, le poids de chaque macro-step de l'ensemble MS retourné par l'algorithme 7 est calculé par les algorithmes 8 et 9.

L'algorithme 8 permet de calculer la matrice $M_{ed} = (m_{i,l})_{\substack{i\in\{1,\cdots,|Q|\}\\ l\in Q}}$ où $m_{i,l}$ représente la probabilité de l'ensemble des chemins partant de l, de longueur i, se terminant par e et ne remettant jamais l'horloge $\psi(source(e))$ à zéro.

Il est polynomial en $O(|Q|^2 \star |\mathcal{E}_{\mathcal{A}}|)$. Enfin, on utilise la matrice M_{ed} dans l'algorithme 9 pour avoir le poids de chaque macro-step de l'ensemble MS. Cet algorithme est linéaire en $|Q|$. Ainsi, nous pouvons conclure que le calcul de l'ensemble des macro-steps selon une transition e et de leurs poids associés est polynomial en $|\mathcal{E}_{\mathcal{A}}|$.

D'autre part, le nombre de points de $\mathcal{A}$ est par définition borné par $|\mathcal{E}_{\mathcal{A}}+1|$. L'algorithme 10 qui permet de calculer l'ensemble $\mathcal{V}$ est donc polynomial en $O(|\mathcal{E}_{\mathcal{A}}+1|^3)$. Le calcul de $\mathcal{V}$ se fait donc en temps polynomial en $|\mathcal{E}_{\mathcal{A}}|$.

Enfin, le temps moyen de convergence vers q_{end} dans $\widehat{MS}(\mathcal{A})$ est calculé en résolvant deux systèmes d'équations linéaires dont chacun est borné par $\mathcal{V}$. Ainsi, le temps moyen de convergence vers q_{end} dans l'automate $ATPPD(\mathcal{A})$ peut se calculer en temps polynomial.

Nous signalons enfin que le degré de complexité de la technique utilisée ne dépend pas exponentiellement du nombre d'horloges ni de la plus grande constante utilisée dans les contraintes temporelles, comme cela est généralement le cas pour les automates temporisés probabilistes [24, 25, 29]. En effet, il n'est pas nécessaire de construire dans cette méthode le système temporisé probabiliste $\mathcal{M}_{\mathcal{A}}$ associé à $\mathcal{A}$ dont la taille est liée au nombre d'horloges et à la plus grande constante de $\mathcal{A}$.

3.2 Temps moyen de convergence dans les ATPPD

Etant donné un ATPPD $\mathcal{A}$ admettant un unique état final absorbant q_{end}, on souhaite calculer le temps moyen de convergence pour atteindre q_{end} en partant de $\bar{q}$. Il sera noté $ExpAbs(\mathcal{A},\kappa)$. Le calcul que nous allons développer est paramétré car il permet d'obtenir une combinaison linéaire E de variables qui sont les paramètres $\mathcal{P}$ de $\mathcal{A}$. En substituant chaque paramètre p apparaissant dans E par sa valeur $\kappa(p)$ dans $\mathbb{N}$, on obtient finalement le temps moyen de convergence $ExpAbs(\mathcal{A},\kappa)$ dans l'ensemble des rationnels $\mathbb{Q}$.

Définition 3.7. *Un macro-chemin est une séquence $\tau = (X_0,q_0) \Rightarrow (X_1,q_1) \Rightarrow \cdots \Rightarrow (X_m,q_m)$, où tous les points (X_i,q_i), $i \in \{0,\cdots,m\}$ appartiennent à $\mathcal{V}$. On dit que la longueur de τ notée $|\tau|$, est égale à m. On définit également le poids de τ, noté $\pi(\tau)$ et la durée de τ, notée $\delta(\tau)$ de la manière suivante :*

$$\pi(\tau) = \prod_{k=0}^{m-1} Wgt((X_k,q_k) \Rightarrow (X_{k+1},q_{k+1}))$$

$$\delta(\tau) = \sum_{k=0}^{m-1} Dur((X_k, q_k) \Rightarrow (X_{k+1}, q_{k+1}))$$

Dans un ATPPD $\mathcal{A}$, $\delta(\tau)$ est une combinaison linéaire des paramètres $\mathcal{P}$ à coefficients dans $\mathbb{N}$ et $\pi(\tau)$ est un nombre rationnel.

Définition 3.8. *Soient (X, q) un point de $\mathcal{A}$ et $MPaths(X, q)$ l'ensemble des macro-chemins de (X, q) à q_{end} dans le graphe des macro-steps associé à $\mathcal{A}$. On définit le coût moyen $MExpAbs(X, q)$ pour atteindre q_{end} dans $\widehat{MS}(\mathcal{A})$ par $\sum_{\tau \in MPaths(X,q)} \delta(\tau) \cdot \pi(\tau)$.*

Remarque 3.5. *Le graphe $\widehat{MS}(\mathcal{A})$ ne répond pas toujours aux propriétés des chaînes de Markov à cause des poids associés aux macro-steps. En effet, la somme des probabilités des chemins issus d'un état (ou point) donné de $\widehat{MS}(\mathcal{A})$ peut être supérieure ou égale à 1, contrairement au cas des chaînes de Markov où elle est égale à 1. Cette différence se traduit au niveau du vecteur W où il est égal au vecteur unité dans le cas d'une chaîne de Markov (voir section 1.6.2).*
Par ailleurs, le calcul de $MExpAbs(X, q)$ correspond à celui d'un automate à coûts positifs comme ceci est décrit dans la section 1.6.2. $MExpAbs(X, q)$ représente ainsi $T(q_{i_0})$ avec $q_{i_0} = (X, q)$.

Dans notre cas, on souhaite calculer le temps moyen de convergence pour atteindre l'état final et absorbant q_{end} en partant de l'état initial $\bar{q}$.

Notation 3.2. *On note $MExpAbs(\mathcal{A})$ le temps moyen de convergence $MExpAbs(\bar{q})$ en partant de l'état initial $\bar{q}$.*

Nous allons démontrer dans ce qui suit l'égalité $ExpAbs(\mathcal{A}, \kappa) = \kappa(MExpAbs(\mathcal{A}))$ qui correspond à la proposition 3.2. Avant d'établir cette égalité, il faut chercher une correspondance entre les chemins de l'ATPPD $\mathcal{A}$ et les macro-chemins du graphe des macro-steps associé à $\mathcal{A}$.

Notation 3.3. *Soient (X, q) et (Y, q') deux points de $\mathcal{V}$. On note $MPaths((X, q), (Y, q'))$ l'ensemble des macro-chemins de (X, q) à (Y, q').*

Etant donné un macro-chemin $\tau \in MPaths(\bar{q}, (X, q))$ et un macro-step $(X, q) \Rightarrow (Y, q')$, on note $\tau \cdot (X, q) \Rightarrow (Y, q')$ le macro-chemin appartenant à $MPaths(\bar{q}, (Y, q'))$ et qui correspond au macro-chemin τ suivi du macro-step $(X, q) \Rightarrow (Y, q')$.

Notation 3.4. *Pour tout point (X, q), on note $Paths(\bar{q}, (X, q))$ l'ensemble des chemins de $Paths(\mathcal{A}, \kappa)$ de la forme*

$$(\bar{q}, 0) \xrightarrow{e_1} q_1 \xrightarrow{e_2} q_2 \cdots \xrightarrow{e_m} q,$$

où X est l'ensemble des horloges remises à zéro par la transition probabiliste e_m.

Etant donné un chemin $\omega \in Paths(\bar{q}, (X, q))$ et un macro-step via σ $(X, q) \overset{\sigma}{\Rightarrow} (Y, q')$, on note $\omega \cdot (X, q) \overset{\sigma}{\Rightarrow} (Y, q')$ le chemin appartenant à $Paths(\bar{q}, (Y, q'))$ qui correspond au chemin ω suivi de l'unique chemin suivant la séquence σ jusqu'au point (Y, q').

Définition 3.9. *Etant donné un point $(X,q) \in \mathcal{V}$ et un macro-chemin $\tau \in MPaths(\bar{q},(X,q))$, l'ensemble des chemins de $\mathcal{A}$ associés à τ noté $Paths(\tau)$, est le sous-ensemble de $Paths(\bar{q},(X,q))$ défini de la manière suivante :*

1. *l'ensemble $Paths(\tau)$ correspond au chemin $\bar{q}$ de longueur nulle si $|\tau| = 0$;*
2. *$Paths(\tau) = \{\omega\cdot(Y,q') \stackrel{\sigma}{\Rightarrow} (X,q) \mid \omega \in Paths(\tau')\ ,\ \sigma \in EndSet((Y,q'),e,(X,q)), e\ \in\ \mathcal{E}_{(X,q)}\}$, si $\tau = \tau' \cdot (Y,q') \Rightarrow (X,q)$, avec $(Y,q') \in \mathcal{V}$, $\tau' \in MPaths(\bar{q},(Y,q'))$ et le macro-step $(Y,q') \Rightarrow (X,q)$.*

Lemme 3.1. *Pour un macro-chemin $\tau \in MPaths(\bar{q},(X,q))$, on a :*

$$\pi(\tau) = \sum_{\omega\in Paths(\tau)} Pr(\omega)$$

$$\kappa(\delta(\tau)\pi(\tau)) = \sum_{\omega\in Paths(\tau)} Pr(\omega)Dur_\kappa(\omega).$$

Démonstration: On démontre cela par réccurrence sur la longueur m du macro-chemin τ. Pour $m = 0$, on a $\pi(\tau) = 1$ et $\sum_{\omega\in Paths(\tau)} Pr(\omega) = Pr(\bar{q}) = 1$ d'où $\pi(\tau) = \sum_{\omega\in Paths(\tau)} Pr(\omega) = 1$. Par ailleurs, $\delta(\tau) = 0$ et $Dur_\kappa(\omega) = Dur_\kappa(\bar{q}) = 0$ pour tout $\kappa \in \Delta$. D'où $\kappa(\delta(\tau)\pi(\tau)) = \sum_{\omega\in Paths(\tau)} Pr(\omega)\ Dur_\kappa\ (\omega)\ =\ 0$.
Soit τ' un macro-chemin de longueur $m+1$, de la forme $\tau \cdot (X,q) \Rightarrow (Y,q')$ où τ est un macro-chemin de longueur m et appartenant à $MPaths(\bar{q},(X,q))$. Ainsi :

$$\begin{aligned}
&\pi(\tau')\\
=\ &\pi(\tau)Wgt((X,q) \Rightarrow (Y,q'))\\
=\ &\sum_{e\in\mathcal{E}_\mathcal{A}} \pi(\tau)Wgt((X,q) \stackrel{e}{\Rightarrow} (Y,q'))\\
=\ &\sum_{e\in\mathcal{E}_\mathcal{A}} \sum_{\sigma\in EndSet((X,q),e,(Y,q'))} \pi(\tau)Wgt((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q'))\\
=\ &\sum_{\omega\in Paths(\tau)} \sum_{e\in\mathcal{E}_\mathcal{A}} \sum_{\sigma\in EndSet((X,q),e,(Y,q'))} Pr(\omega)Wgt((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q'))\\
=\ &\sum_{\omega'\in Paths(\tau')} Pr(\omega'),
\end{aligned}$$

De plus, on a :

$$\begin{aligned}
&\pi(\tau')\delta(\tau')\\
=\ &\pi(\tau)Wgt((X,q) \Rightarrow (Y,q'))(\delta(\tau) + Dur((X,q) \Rightarrow (Y,q')))\\
=\ &\pi(\tau)Wgt((X,q) \Rightarrow (Y,q')) \left(\delta(\tau) + \sum_{e\in\mathcal{E}_\mathcal{A}} \frac{Wgt((X,q) \stackrel{e}{\Rightarrow} (Y,q')) \cdot Dur((X,q) \stackrel{e}{\Rightarrow} (Y,q'))}{Wgt((X,q) \Rightarrow (Y,q'))}\right)\\
=\ &\pi(\tau)Wgt((X,q) \Rightarrow (Y,q'))\delta(\tau) + \pi(\tau) \sum_{e\in\mathcal{E}_\mathcal{A}} Wgt((X,q) \stackrel{e}{\Rightarrow} (Y,q'))Dur((X,q) \stackrel{e}{\Rightarrow} (Y,q'))\\
=\ &\sum_{e\in\mathcal{E}_\mathcal{A}} \pi(\tau)Wgt((X,q) \stackrel{e}{\Rightarrow} (Y,q'))(\delta(\tau) + Dur((X,q) \stackrel{e}{\Rightarrow} (Y,q')))\\
=\ &\sum_{e\in\mathcal{E}_\mathcal{A}} \sum_{\sigma\in EndSet((X,q),e,(Y,q'))} \pi(\tau)Wgt((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q'))(\delta(\tau) + Dur((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q')))\\
=\ &\sum_{\omega\in Paths(\tau)} \sum_{e\in\mathcal{E}_\mathcal{A}} \sum_{\sigma\in EndSet((X,q),e,(Y,q'))} Pr(\omega)Wgt((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q'))(\delta(\tau) + Dur((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q'))),
\end{aligned}$$

où la dernière égalité provient de la première équation de ce lemme, $\pi(\tau) = \sum_{\omega \in Paths(\tau)} Pr(\omega)$. Pour tout $\omega' \in Paths(\tau')$, on a $\kappa(\delta(\tau) + Dur((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q'))) = Dur_\kappa(\omega')$, et ainsi :

$$\sum_{\omega \in Paths(\tau)} \sum_{e \in \mathcal{E}_{(Y,q')}} \sum_{\sigma \in EndSet((X,q),e,(Y,q'))} Pr(\omega) Wgt((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q'))(\delta(\tau) + Dur((X,q) \stackrel{\sigma}{\Rightarrow} (Y,q')))$$
$$= \sum_{\omega' \in Paths(\tau')} Pr(\omega') Dur_\kappa(\omega').$$

□

Lemme 3.2. *Pour tout $(X,q) \in \mathcal{V}$, on a $Paths(\bar{q},(X,q)) = \cup_{\tau \in MPaths(\bar{q},(X,q))} Paths(\tau)$. La réunion est disjointe car la représentation est unique.*

Démonstration: Par définition, pour tout point (X,q), on a $Paths(\bar{q},(X,q)) \supseteq \cup_{\tau \in MPaths(\bar{q},(X,q))} Paths(\tau)$. Il reste à montrer que pour tout point $(X,q) \in \mathcal{V}$, tout chemin $\omega \in Paths(\bar{q},(X,q))$ appartient à $Paths(\tau)$ pour un $\tau \in MPaths(\bar{q},(X,q))$. Montrons cela par réccurrence sur la longueur de ω. Si ω est de longueur nulle alors $\omega \in Paths(\tau)$ avec $\tau = \bar{q}$ le macro-chemin est de longueur nulle. Supposons maintenant que ω est de longueur m non nulle et de la forme $(\bar{q}, \mathbf{0}) \stackrel{e_1}{\to} q_1 \stackrel{e_2}{\to} \cdots \stackrel{e_m}{\to} q$ où $\mathbf{0}$ est la valuation qui remet toutes les horloges à zéro. Soit $e_k = (q_{k-1}, Y_k, q_k)$, $k \leq m$, la dernière transition probabiliste qui remet $\psi(q_{m-1})$ à zéro dans la séquence $e_1 e_2 \cdots e_{m-1}$.
Si $k = 0$, ceci correspond à la remise à zéro de toutes les horloges lorsque l'on entre en $\bar{q}$. On obtient le macro-step $(\mathcal{X}, \bar{q}) \stackrel{\sigma}{\Rightarrow} (X,q)$ avec $\sigma = e_1 \cdots e_m$. Donc $\omega \in Paths(\tau)$ avec $\tau = (\mathcal{X}, \bar{q}) \Rightarrow (X,q)$.
Si $k \neq 0$, le chemin ω est de la forme $\omega' \cdot ((Y_k, q_k) \stackrel{\sigma}{\Rightarrow} (X,q))$ avec $(Y_k, q_k) \in \mathcal{V}$, $\omega' \in Paths(\bar{q},(Y_k,q_k))$ et $\sigma = e_{k+1} \cdots e_m$ la séquence qui définit le macro-step $(Y_k,q_k) \stackrel{\sigma}{\Rightarrow} (X,q)$. D'après l'hypothèse de récurrence, $\omega' \in Paths(\tau')$ pour un unique $\tau' \in MPaths(\bar{q},(Y_k,q_k))$. Ainsi, $\omega \in Paths(\tau' \cdot (Y_k,q_k) \stackrel{\sigma}{\Rightarrow} (X,q))$. Le macro-chemin τ est donc égal à $\tau' \cdot (Y_k,q_k) \stackrel{\sigma}{\Rightarrow} (X,q)$ et appartient donc à $MPaths(\bar{q},(X,q))$. On conclut que $\omega \in Paths(\tau)$.

□

Nous déduisons des lemmes 3.1 et 3.2 que le temps moyen de convergence vers l'état final et absorbant q_{end} dans un ATPPD $\mathcal{A}$ muni d'une valuation paramétrique κ est égal au temps moyen de convergence vers q_{end} dans le graphe des macro-steps $ATPPD(\mathcal{A})$ en y substituant chaque paramètre par sa valeur dans $\mathbb{N}$.

Proposition 3.2. *Soient $\mathcal{A}$ un ATPPD muni d'une valuation paramétrique κ. On a $ExpAbs(\mathcal{A},\kappa) = \kappa(MExpAbs(\mathcal{A}))$.*

Démonstration: On a

$$\begin{aligned} & ExpAbs(\mathcal{A},\kappa) \\ = & \sum_{\omega \in \Omega(q_{end})} Pr(\omega) Dur_\kappa(\omega) \\ = & \sum_{\tau \in MPaths(\bar{q},q_{end})} \sum_{\omega \in Paths(\tau)} Pr(\omega) Dur_\kappa(\omega) \end{aligned}$$

d'après le lemme 3.2. Par le lemme 3.1, on a :

$$\begin{aligned}
&\sum_{\tau \in MPaths(\bar{q}, q_{end})} \sum_{\omega \in Paths(\tau)} Pr(\omega) Dur_{\kappa}(\omega) \\
&= \sum_{\tau \in MPaths(\bar{q}, q_{end})} \kappa(\delta(\tau)\pi(\tau)) \\
&= \kappa(\sum_{\tau \in MPaths(\bar{q}, q_{end})} \delta(\tau)\pi(\tau)) \\
&= \kappa(MExpAbs(\mathcal{A}))
\end{aligned}$$

□

3.3 Calcul du coût moyen de convergence sur les graphes des macro-steps

Nous allons présenter par la suite une méthode de calcul de $MExpAbs(\mathcal{A})$. On pose $\mathcal{V}' = \mathcal{V} \setminus q_{end}$. Nous ordonnons les éléments de $\mathcal{V}$ en mettant le point q_{end} comme dernier élément. Il n'y a pas de règles à imposer sur l'ordre que l'on choisit de donner aux autres points.

Définition 3.10. *On définit la matrice carrée* $\mathbf{M}$ *sur l'ensemble* $\mathcal{V}'$ *par :*
$\mathbf{M}((X,q),(Y,q')) = Wgt((X,q) \Rightarrow (Y,q'))$ *pour tout* (X,q), $(Y,q') \in \mathcal{V}'$ *tel que* $(X,q) \Rightarrow (Y,q')$ *et* $\mathbf{M}((X,q),(Y,q')) = 0$ *sinon.*

Lemme 3.3. *La matrice* $I - \mathbf{M}$ *est inversible. Son inverse est la limite de la série convergente* $\sum_{k=0}^{m} \mathbf{M}^k$.

Démonstration: On considère la chaîne de Markov $Untimed(\mathcal{A})$ obtenue à partir de $\mathcal{A}$ en écartant les données temporelles tels les gardes et les invariants. Montrons que la série $U_m = \sum_{k=0}^{m} \mathbf{M}^k$ est convergente.
Pour tout entier m et tout point (X,q), $(Y,q') \in \mathcal{V}'$, $(\mathbf{M})^k((X,q),(Y,q'))$ est le poids du macro-chemin de (X,q) à (Y,q') dont chacun est de la forme :

$$(X,q) \Rightarrow (X_1,q_1) \Rightarrow (X_2,q_2) \Rightarrow \cdots \Rightarrow (X_{k-1},q_{k-1}) \Rightarrow (Y,q').$$

$(\mathbf{M})^k((X,q),(Y,q'))$ est donc égal à la probabilité de l'ensemble des chemins dans $Untimed(\mathcal{A})$ représentés par les suites de macro-steps de longueur k d'après le lemme 3.2 et qui sont de la forme :

$$q \rightarrow \cdots \stackrel{e_1}{\rightarrow} q_1 \rightarrow \cdots \stackrel{e_2}{\rightarrow} q_2 \cdots \stackrel{e_{k-1}}{\rightarrow} q_{k-1} \rightarrow \cdots \stackrel{e_k}{\rightarrow} q'.$$

Ainsi, U_m est la somme des probabilités de tous les chemins de q à q' dans $Untimed(\mathcal{A})$ qui sont représentés par des macro-chemins de (X,q) à (Y,q') de longueur inférieure ou égale à m.
Etant donné que $Untimed(\mathcal{A})$ est une chaîne de Markov, la somme des probabilités de tous les chemins de q à q' est inférieure ou égale à 1.
On conclut que la suite des sommes partielles $(U_m)_m$ est bornée par 1. Elle est de plus croissante car la matrice $\mathbf{M}$ est à termes positifs. Donc la série $\sum_{k=0}^{\infty} \mathbf{M}^k((X,q)(Y,q'))$ est convergente.
La matrice $I - \mathbf{M}$ est donc inversible et son inverse est $\sum_{k=0}^{\infty} \mathbf{M}^k$.

□

Soit $\mathcal{A}$ un ATPPD. Le calcul du temps moyen de convergence dans le graphe des macro-steps $\widehat{MS}(\mathcal{A})$ fait appel à la formule de Bertsekas développée dans la section 1.6.2.

Pour tout point $(X, q) \in \mathcal{V}'$, on introduit un *facteur correcteur* noté $w(X, q)$.

Définition 3.11. *On définit le facteur correcteur $w(X, q)$ comme égal au poids de l'ensemble des macro-chemins de (X, q) à q_{end} :*

$$w(X, q) = \sum_{\tau \in MPaths(X,q)} \pi(\tau).$$

Comme nous l'avons déjà mentionné dans la section 1.6.2, le facteur correcteur est égal à 1 si le graphe des macros-steps $\widehat{MS}(\mathcal{A})$ est une chaîne de Markov. Dans le cas contraire, il peut être supérieur ou égal à 1.

Nous pouvons aussi donner une définition du facteur correcteur équivalente à celle décrite dans la définition 3.11. Celle-ci exprime le facteur correcteur en tout point $(X, q) \in \mathcal{V}'$ comme une limite du facteur correcteur w^m défini sur les macro-chemins de longueur finie, inférieure ou égale à m. On a :

$$w^m(X, q) = \sum_{\tau \in MPaths^{\leq m}(X,q)} \pi(\tau).$$

Ainsi,

$$w(X, q) = \lim_{m \to \infty} w^m(X, q).$$

La limite de la suite $w^m(X, q)$ est bien définie car $w^m(X, q)$ est positive et croissante.

Lemme 3.4. *Soit $W = (w(X, q))_{(X,q)\in\mathcal{V}'}$. Soit B le vecteur de dimension $|\mathcal{V}'|$ égal à $(Wgt((X, q) \Rightarrow q_{end}))_{(X,q)\in\mathcal{V}'}$. Le vecteur W est l'unique solution du système $W = \mathbf{M}W + B$.*

Démonstration: Nous utiliserons la deuxième définition du facteur correcteur.

D'après le lemme 3.3, la matrice $I - \mathbf{M}$ est inversible. Donc, le système $Z = \mathbf{M}Z + B$ admet une unique solution. Montrons que W est cette unique solution.

Soit $(X, q) \in \mathcal{V}'$, on a :

$$\begin{aligned}
& w^{m+1}(X, q) \\
&= \sum_{\tau \in MPaths^{\leq m+1}((X,q),q_{end})} \pi(\tau) \\
&= \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X, q), (Y, q')) \sum_{\tau \in MPaths^{\leq m}((Y,q'),q_{end})} \pi(\tau) + Wgt((X, q) \Rightarrow q_{end}) \\
&= \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X, q), (Y, q'))w^m(Y, q') + Wgt((X, q) \Rightarrow q_{end}) \\
&= \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X, q), (Y, q'))w^m(Y, q') + B(X, q).
\end{aligned}$$

En prenant la limite dans les deux termes de l'égalité quand m tend vers l'infini, on obtient :

$$w(X,q) = \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X,q),(Y,q'))w(Y,q') + B(X,q).$$

Donc $W = (w(X,q))_{(X,q)\in\mathcal{V}'}$ est l'unique solution du système avec $W = \mathbf{M}Z + B$.

□

Pour tout point (X,q), $(Y,q') \in \mathcal{V}'$, on pose :
– $\zeta((X,q),(Y,q')) = Dur((X,q) \Rightarrow (Y,q')) \cdot \mathbf{M}((X,q),(Y,q')) \cdot w(Y,q')$
– $\eta(X,q) = Dur((X,q) \Rightarrow q_{end}) \cdot Wgt((X,q) \Rightarrow q_{end})$

Proposition 3.3. *Soit C le vecteur de dimension $|\mathcal{V}'|$ égal à :*

$$\left(\sum_{(Y,q')\in\mathcal{V}'} \zeta((X,q),(Y,q')) + \eta(X,q) \right)_{(X,q)\in\mathcal{V}'}$$

Le vecteur $T = (MExpAbs(X,q))_{(X,q)\in\mathcal{V}'}$ est l'unique solution du système $T = \mathbf{M}T + C$.

Démonstration: On donne tout d'abord une définition équivalente à T comme suit. Pour tout $(X,q) \in \mathcal{V}'$, on a $T(X,q) = \lim_{m\to\infty} T^m(X,q)$ où

$$T^m(X,q) = \sum_{\tau\in MPaths^{\leq m}((X,q),q_{end})} \delta(\tau)\pi(\tau).$$

La limite de $T^m(X,q)$ est bien définie car la suite $T^m(X,q)$ est positive et croissante.

D'après le lemme 3.3, $I - \mathbf{M}$ est inversible, donc le système $Z = \mathbf{M}Z + C$ admet une unique solution. Montrons que T est cette unique solution.

Soit $(X,q) \in \mathcal{V}'$, on a :

$$\begin{aligned}
&T^{m+1}(X,q)\\
&= \sum_{\tau\in MPaths^{\leq m+1}((X,q),q_{end})} \delta(\tau)\pi(\tau)\\
&= \sum_{(Y,q')\in\mathcal{V}'} \sum_{\tau\in MPaths^{\leq m}((Y,q'),q_{end})} (\delta(\tau) + Dur((X,q) \Rightarrow (Y,q')))\mathbf{M}((X,q),(Y,q'))\pi(\tau)\\
&+ Dur((X,q) \Rightarrow q_{end})Wgt((X,q) \Rightarrow q_{end})\\
&= \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X,q),(Y,q')) \sum_{\tau\in MPaths^{\leq m}((Y,q'),q_{end})} \delta(\tau)\pi(\tau)\\
&+ \sum_{(Y,q')\in\mathcal{V}'} Dur((X,q) \Rightarrow (Y,q')))\mathbf{M}((X,q),(Y,q')) \sum_{\tau\in MPaths^{\leq m}((Y,q'),q_{end})} \pi(\tau)\\
&+ Dur((X,q) \Rightarrow q_{end})Wgt((X,q) \Rightarrow q_{end})\\
&= \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X,q),(Y,q'))T^m(Y,q')\\
&+ \sum_{(Y,q')\in\mathcal{V}'} Dur((X,q) \Rightarrow (Y,q')))\mathbf{M}((X,q),(Y,q'))w^m(Y,q')\\
&+ Dur((X,q) \Rightarrow q_{end})Wgt((X,q) \Rightarrow q_{end}).
\end{aligned}$$

En prenant la limite dans les deux termes de l'égalité quand m tend vers l'infini, on obtient :

$$
\begin{aligned}
& T(X,q) \\
= & \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X,q),(Y,q'))T(Y,q') \\
+ & \sum_{(Y,q')\in\mathcal{V}'} Dur((X,q)\Rightarrow(Y,q')))\mathbf{M}((X,q),(Y,q'))w(Y,q') \\
+ & \ Dur((X,q)\Rightarrow q_{end})Wgt((X,q)\Rightarrow q_{end}).
\end{aligned}
$$

Ainsi,

$$T(X,q) = \sum_{(Y,q')\in\mathcal{V}'} \mathbf{M}((X,q),(Y,q'))T(Y,q') + C(X,q),$$

avec

$$C(X,q) = \sum_{(Y,q')\in\mathcal{V}'} Dur((X,q)\Rightarrow(Y,q'))\mathbf{M}((X,q),(Y,q'))w(Y,q') + Dur((X,q)\Rightarrow q_{end})Wgt((X,q)\Rightarrow q_{end}).$$

□

Remarque 3.6. *La proposition 3.3 permet de calculer le temps moyen de convergence $MExpAbs(\mathcal{A})$ qui est égal à la première composante $T[0] = MExpAbs(\bar{q})$ de la solution T. $MExpAbs(\mathcal{A})$ est donné sous forme d'une combinaison linéaire γ de l'ensemble des paramètres $\mathcal{P}$ de $\mathcal{A}$ car le vecteur C fait intervenir les durées des macro-steps exprimées chacune par un paramètre. D'après la proposition 3.2, on déduit du calcul de T le temps moyen de convergence vers q_{end} dans l'ATPPD $\mathcal{A}$ car $ExpAbs(\mathcal{A},\kappa) = \kappa(MExpAbs(\mathcal{A}))$. On remplace ainsi chaque paramètre apparaissant dans γ par sa valeur dans $\mathbb{N}$.*

Exemple 3.5. *Prenons le graphe des macro-steps décrit dans la figure 3.5. L'ensemble $\mathcal{V}' = \{(\mathcal{X},q_1),(\{x,y\},q_1),(\{z\},q_3)\}$. Soit I la matrice identité de dimension $|\mathcal{V}'|$. Le vecteur $B = (B_{(\mathcal{X},q_1)}, B_{(\{x,y\},q_1)}, B_{(\{z\},q_3)})$ est égal à :*

- $B_{(\mathcal{X},q_1)} = Wgt((\mathcal{X},q_1)\Rightarrow q_{end}) = 0$,
- $B_{(\{x,y\},q_1)} = Wgt((\{x,y\},q_1)\Rightarrow q_{end}) = 0$,
- $B_{(\{z\},q_3)} = Wgt((\{z\},q_3)\Rightarrow q_{end}) = \beta$,

Le vecteur W est égal à $(I-\mathbf{M})^{-1}B = (1,1,\beta)$.
Par ailleurs, le vecteur $C = (C_{(\mathcal{X},q_1)}, C_{(\{x,y\},q_1)}, C_{(\{z\},q_3)})$ est égal à :

- $C_{(\mathcal{X},q_1)} = T1\cdot\mathbf{M}((\mathcal{X},q_1),(\{x,y\},q_1))\cdot 1 + TD\cdot\mathbf{M}((\mathcal{X},q_1),(\{z\},q_3))\cdot\beta = (1-\alpha\beta)T1 + \alpha\beta TD$,
- $C_{(\{x,y\},q_1)} = T1\cdot\mathbf{M}((\{x,y\},q_1),(\{x,y\},q_1))\cdot 1 + TD\cdot\mathbf{M}((\{x,y\},q_1),(\{z\},q_3))\cdot\beta = (1-\alpha\beta)T1 + \alpha\beta TD$,
- $C_{(\{z\},q_3)} = TD\cdot Wgt((\{z\},q_3)\Rightarrow q_{end}) = \beta TD$.

Enfin, le vecteur $T = (T_{(\mathcal{X},q_1)}, T_{(\{x,y\},q_1)}, T_{(\{z\},q_3)})$ est donné par $(I-\mathbf{M})^{-1}C$. Il est égal à :

- $T_{(\mathcal{X},q_1)} = \frac{1-\alpha\beta}{\alpha\beta}T1 + 2TD$,
- $T_{(\{x,y\},q_1)} = \frac{1-\alpha\beta}{\alpha\beta}T1 + 2TD$,
- $T_{(\{z\},q_3)} = \beta TD$.

Etant donné que $MExpAbs(\mathcal{A}) = MExpAbs(\bar{q})$ tel que $\bar{q} = (\mathcal{X},q_1)$, on conclut que $MExpAbs(\mathcal{A}) = \frac{1-\alpha\beta}{\alpha\beta}T1 + 2TD$. D'après la proposition 3.2, nous pouvons déduire $ExpAbs(\mathcal{A},\kappa)$.
En effet, l'ensemble $\Delta \subseteq \mathbb{N}^{\mathcal{P}}$ vu dans l'exemple 2.8 montre que pour tout $\kappa\in\Delta$, $\kappa(T1) > 2\kappa(TD)$. Soit une valuation paramétrique κ telle que $\kappa(T1) = 5$ et $(TD) = 2$. $ExpAbs(\mathcal{A},\kappa)$ sera donc égal à $5\cdot\frac{1-\alpha\beta}{\alpha\beta} + 4$ sachant que α et β représentent des probabilités dans $\mathbb{Q}$.

3.4 Application : le protocole CSMA/CD

Cette section sera consacrée à l'application de la méthode développée dans la section précédente. Le protocole $CSMA/CD$ (en anglais, Carrier Sense Multiple Access/Collision Detection) est utilisé pour la transmission de données dans les réseaux Ethernet. Ce protocole sera modélisé de manière probabiliste dans le but d'éviter au maximum les collisions entre les messages. La modélisation adoptée dans la suite est celle proposée dans les articles [27, 28, 1].

3.4.1 Description du protocole

Le protocole $CSMA/CD$ est un protocole qui gère les flux de communications entre plusieurs stations (qu'on appellera parfois émetteurs), qui dialoguent à travers un canal unique [27, 28]. Toutes les stations peuvent de manière identique envoyer des messages sur le réseau (Multiple Access). Dans notre cas, nous nous restreindrons au cas de deux émetteurs S_1 et S_2 et le canal C.

Chaque émetteur doit tout d'abord écouter le canal pour savoir s'il est libre ou occupé (Carrier Sense) et attendre que le canal apparaisse comme libre pour envoyer son message. Si le canal est occupé, l'émetteur doit attendre jusqu'à ce que le canal se libère. Si les deux émetteurs envoient leurs messages au même instant, une collision a lieu et un message de collision est envoyé (Collision Detection) aux émetteurs. Le temps d'attente avant réémission est choisi par chaque émetteur de manière aléatoire dans un ensemble I.

Le temps de propagation d'un message dans le canal entre les deux émetteurs est égal à σ unités de temps et le temps d'émission d'un message complet est égal à λ unités de temps. σ et λ seront dans notre cas des paramètres. Soit $\mathcal{P}$ l'ensemble $\{\sigma, \lambda\}$.

La modélisation du protocole est obtenue par la composition parallèle de trois automates : les deux émetteurs et le canal. L'automate modélisant un émetteur est un automate temporisé où le choix du temps d'attente dans l'ensemble I avant réémission se fait de manière non déterministe. L'automate qui modélise le canal est un automate temporisé où le non déterminisme apparaît au niveau du temps nécessaire pour que le canal se libère après qu'une collision n'ait lieu.

3.4.1.1 Le canal

Le canal C décrit dans la figure 3.6 est initialement libre et aucun message n'est en cours de transmission. Lorsque l'un des émetteurs S_1 (resp. S_2) décide d'envoyer un message $send_1$ (resp. $send_2$), le canal se met en état $Transmit_c$. Il faut au plus σ unités de temps depuis le début de l'émission pour que le message se propage entre les deux stations. Pendant ces σ unités de temps, S_2 (resp. S_1) peut émettre car il ne détecte pas encore la présence d'un message dans le canal. Une collision a donc lieu cd entre S_1 et S_2 dans le cas où S_2 (resp. S_1) émet et le canal se met en état $Collide_c$. Si les σ unités de temps sont écoulées sans que S_2 (resp. S_1) ne transmette de message, le canal est vu comme occupé par S_2 (resp. S_1) $busy_2$ (resp. $busy_1$). A la fin de la transmission du message par S_1 (resp. S_2) end_1 (resp. end_2), le canal revient en position $Init_c$. En cas de collision, il faut attendre au plus σ unités de temps avant que le canal ne soit de nouveau libre et qu'il ne passe à l'état $Init_c$.

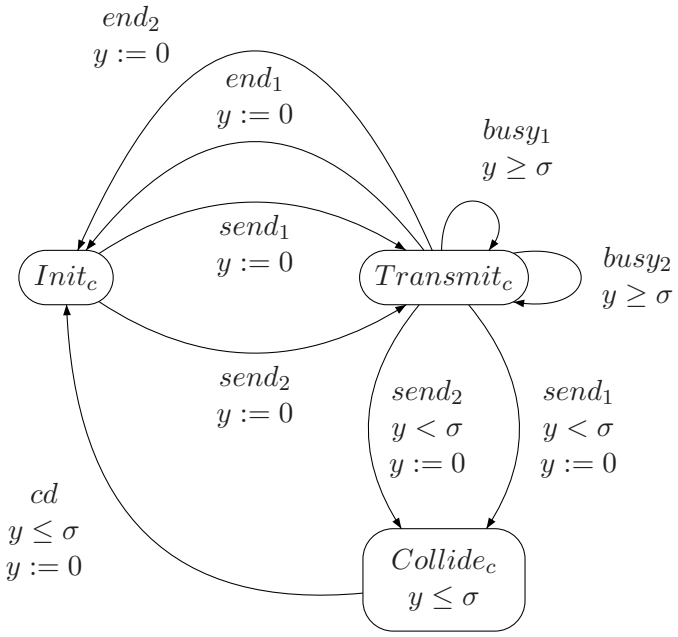

FIG. 3.6 – Canal modélisant le canal du protocole CSMA/CD

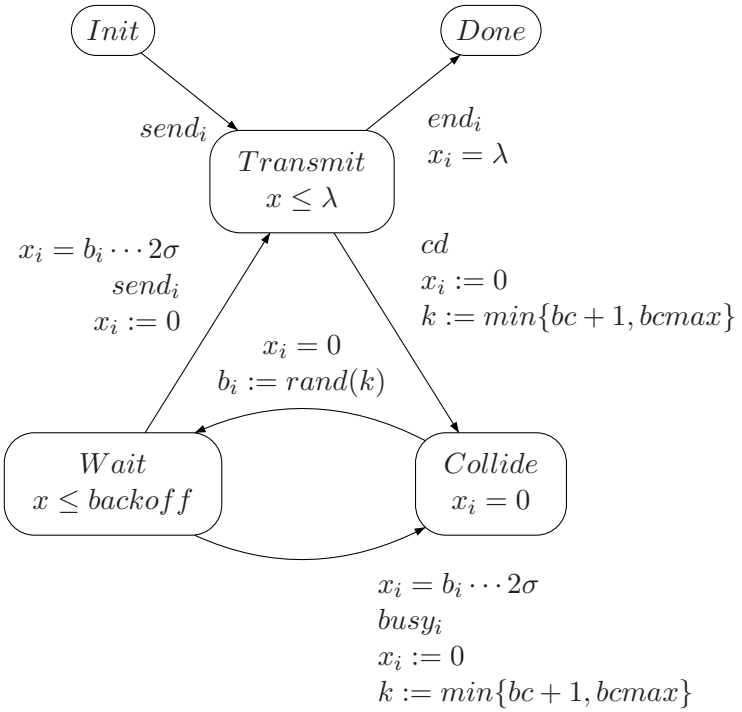

FIG. 3.7 – Automate modélisant l'émetteur S_i du protocole CSMA/CD

3.4.1.2 Les émetteurs

Un émetteur S_i décrit dans la figure 3.7 commence à envoyer un message ($send_i$). Si aucune collision n'a lieu, le message est entièrement émis au bout de λ unités de temps (end_i). En cas de collision *cd*, l'émetteur attend pendant une durée $2\sigma \cdot b_i$. L'entier b_i, appelé $backoff$, est à chaque fois (i.e à chaque passage dans l'état $Collide$) choisi aléatoirement dans l'ensemble $\{0, 1, 2, \cdots, 2^{k+1} - 1\}$ où $k = min(bc, bcmax)$ avec *bc* le nombre total de collisions et *bcmax* une constante du protocole. Au bout de $2\sigma \cdot b_i$ unités de temps, l'émetteur réémet son message s'il sent que le canal est libre, sinon il retourne en situation $Collide$ et attend de nouveau pendant $2\sigma \cdot b_i$ unités de temps.

Après une collision, deux cas se profilent :

- Si $b_1 = b_2$, S_1 et S_2 réémettent leurs messages au même instant. Une nouvelle collision se produit et le nombre *bc* est incrémenté de 1.
- Si $b_1 \neq b_2$, l'émetteur ayant le $backoff$ le plus petit peut émettre son message entièrement car le deuxième émetteur peut décider d'envoyer un message après au moins 2σ unités de temps du premier, ce qui lui permettra de détecter que le canal est occupé (en rappelant qu'il faut σ unités de temps pour qu'un message se propage entre deux stations).

3.4.1.3 Contraintes sur les paramètres du protocole

Avant d'effectuer la composition parallèle des automates de S_1, S_2 et C, on remarque que le cas où $\lambda < 2\sigma$ peut mener à des comportements non souhaités du système.

En effet, le temps maximal nécessaire pour qu'un message émis par un des émetteurs ne se propage dans le canal et qu'un message de notification ne soit envoyé aux émetteurs par le canal en cas de collision est égal à 2σ. Si la longueur du message à émettre est strictement inférieure à 2σ, le message de notification en cas de collision pourrait être reçu par l'émetteur en question après l'émission total de son message. L'émetteur ne tient donc pas compte de la collision qui a eu lieu alors que le canal l'a détectée. Le processus normal que doit suivre l'ensemble du système en cas de collision ne peut plus ainsi se faire car l'émetteur en question ne choisit pas de $backoff$ pour retransmettre ultérieurement son message. On peut donc supposer que $\lambda \geq 2\sigma$.

Au niveau du modèle, le temps est arrêté car la synchronisation entre les composants ne peut plus se faire.

Supposons que S_1 envoie un message à l'instant $t = 0$ et que S_2 envoie un message à l'instant $t' < \sigma$. Une collision a donc lieu dans le canal, qui passe alors dans l'état $Collide_c$. Il faut au plus σ unités de temps pour que le canal avertisse les émetteurs de l'existence d'une collision et qu'il revienne en position $Init_c$. Si le premier message se termine à l'instant $t'' < 2\sigma$, le canal est encore en position $Collide_c$. Le système entier est dans l'état $Transmit_1 \cdot Transmit_2 \cdot Collide_c$. La garde $x_1 = \lambda$ de la transition end_1 est vérifiée mais end_1 ne peut pas être tirée car le canal est en position $Collide_c$. Le temps ne peut donc plus passer. Cette situation de blocage temporel traduit un comportement zenon (voir la section 2.1.2).

Pour éviter cela, on impose la condition $\lambda \geq 2\sigma$ sur les paramètres du protocole σ et λ.

3.4.1.4 Automate produit

L'automate produit est obtenu en faisant la composition synchrone des automates de S_1, S_2 et C. L'état initial de l'automate produit traduit un premier état de collision : le nombre de collision *bc* est égal à 1 et l'automate est dans l'état $Collide_c \cdot Transmit_1 \cdot Transmit_2$.

Après au plus σ unités de temps, on est dans l'état $Init_c \cdot Collide_1 \cdot Collide_2$ équivalent à l'état $Init_c \cdot Wait_1 \cdot Wait_2$ car le choix du $backoff$ se fait instantanément. Chaque émetteur doit choisir un $backoff$ dans l'ensemble $\{0, 1, 2, 3\}$ car $k = min(1, 2) = 1$. Si $bc_1 = bc_2 = m$ (cd_{A1}), une deuxième collision a lieu ($send_1 \cdot send_2$) et bc devient égal à 2. Dans le cas où $bc_1 = m_1$ et $bc_2 = m_2$ (cd_{B1}) tels que $m_1 \neq m_2$, on suppose par symétrie que $m_1 < m_2$. S_1 peut alors envoyer un message complet ($send_1 \cdot end_1$).

Dans le cas de la deuxième collision, S_1 et S_2 doivent de nouveau choisir un $backoff$ mais cette fois-ci dans l'ensemble $\{0, 1, 2, 3, \cdots, 7\}$ car $k = min(2, 2) = 2$. Si $bc_1 = bc_2 = n$ (cd_{A2}), une troisième collision a lieu ($send_1 \cdot send_2$) et bc devient égal à 2. Dans le cas où $bc_1 = n_1$ et $bc_2 = n_2$ (cd_{B2}) tels que $n_1 \neq n_2$, on suppose par symétrie que $n_1 < n_2$. S_1 peut alors envoyer un message complet ($send_1 \cdot end_1$).

Pour $bc \geq 3$, les scénarios qui suivent sont identiques au cas $bc = 2$ car k se stabilise à 2. La modélisation que nous proposons à travers cet automate produit décrit dans la figure 3.8 se limite à décrire le fonctionnement du protocole en partant d'une situation de première collision (état q_1) jusqu'à ce que l'un des émetteurs émette un premier message avec succès (état q_{end}).

3.4.1.5 L'ATPPSD de CSMA/CD

Dans l'automate produit de la figure 3.8, on remarque que le choix du $backoff$ par les émetteurs se fait de manière non déterministe au niveau de l'état q_1 (transitions cd_{A1} et cd_{B1}) et de l'état q_3 (transitions cd_{A2} et cd_{B2}).

Dans le but de supprimer ce non déterminisme, nous le remplaçons par un choix probabiliste uniforme au niveau des états q_1 et q_3.

Dans l'état q_1, on choisit de façon uniforme entre l'ensemble des transitions étiquetées par cd_{A1} et l'ensemble des transitions étiquetées par cd_{B1}. De l'état q_1, on peut ainsi choisir de tirer une transition de l'un de ces deux ensembles avec une probabilité égale à 1/2. La même explication s'applique à l'état q_3 et aux deux ensembles de transitions étiquetées respectivement par cd_{A2} et cd_{B2}.

On obtient ainsi 2 distributions :
- $prob(q_1)$ avec $prob(q_1)(q_{(2_=, m=j)}) = 1/16$ pour tout $j \in \{0, 1, 2, 3\}$ et $prob(q_1)(q_{(2_{\neq}, (m_1=i, m_2=j))}) = 1/8$ pour tout $i \in \{0, \cdots, 2\}$, $j \in \{1, \cdots, 3\}$ et $m_1 < m_2$.
- $prob(q_3)$ avec $prob(q_3)(q_{(2'_=, n=k)}) = 1/64$ pour tout $k \in \{0, \cdots, 7\}$ et $prob(q_3)(q_{(2'_{\neq}, (n_1=k, n_2=l))}) = 1/32$ pour tout $k \in \{0, \cdots, 6\}$, $l \in \{1, \cdots, 7\}$ et $n_1 < n_2$.

En ayant ainsi remplacé le choix non déterministe dans le choix des $backoff$ par un choix uniforme probabiliste dans l'automate produit décrit dans la figure 3.8, on obtient un Automate Temporisé Probabiliste Paramétré Semi Déterminé (ATPPSD) noté $\mathcal{A}$.

$\mathcal{A}$ bien formé et non zenon.

Nous devons maintenant déterminer un ensemble de valuations paramétriques $\Delta \subseteq \mathbb{N}^{\mathcal{P}}$ tel que pour tout $\kappa \in \Delta$, $\mathcal{A}_\kappa$ soit bien formé (voir la section 2.5.1.2) et fortement non zenon (voir la section 2.5.1.3).

D'après la section 3.4.1.3, la spécification du protocole impose la contrainte $\lambda \geq 2\sigma$ sur l'ensemble des paramètres $\mathcal{P}$ de l'ATPPSD $\mathcal{A}$ pour éviter les comportements zenon.

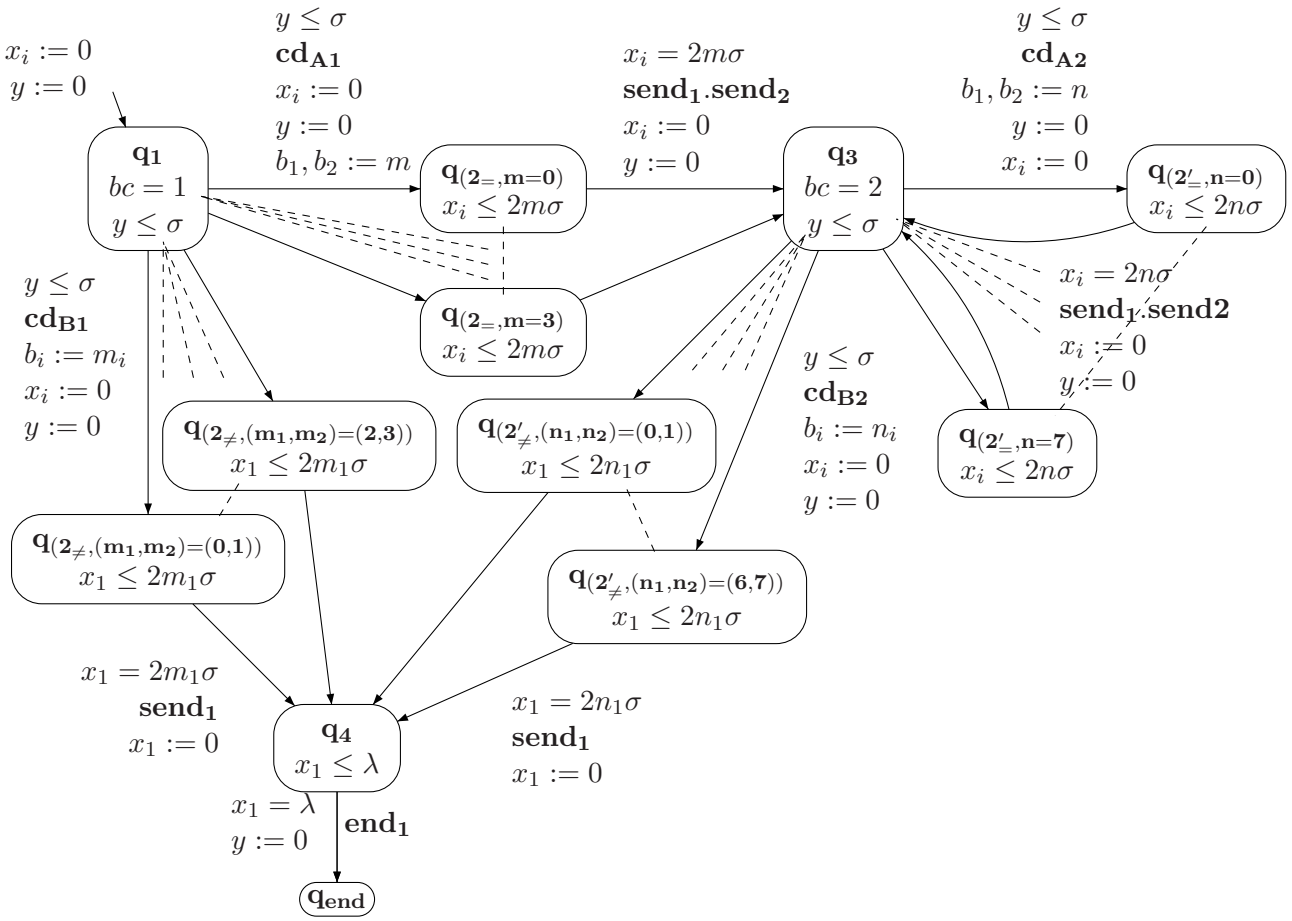

FIG. 3.8 – Automate produit du protocole CSMA/CD

Montrons que $\mathcal{A}$ est bien formé et fortement non zenon pour l'ensemble Δ de valuations paramétriques tel que pour tout $\kappa \in \Delta$, $\kappa(\lambda) \geq 2\kappa(\sigma)$.

$\mathcal{A}$ bien formé.

Comme nous l'avons vu dans la section 2.5.1.2, un ATPPSD est bien formé pour l'ensemble Δ si l'état puit q_{sink} de l'automate temporisé paramétré équivalent à $\mathcal{A}$ n'est pas accessible. Cette propriété peut être vérifiée grâce à l'outil HyTech [20].

Dans notre cas, nous pouvons vérifier par inspection que $\mathcal{A}$ est bien formé pour tout $\kappa \in \Delta$.

Pour tout état q de $\mathcal{A}$, on peut facilement voir que pour toute valuation ν telle que $\langle q, \nu \rangle$ soit accessible dans $\mathcal{A}_\kappa$, on a $\nu \in free_\kappa(q)$. Pour chaque transition, l'horloge testée pour l'invariant de l'état d'arrivée est remise à zéro par la transition. En effet, on constate que pour toute transition probabiliste $e = (q, prob(q)(q'), \psi(q) \leq \phi(q'), X, q')$ issue de q on a

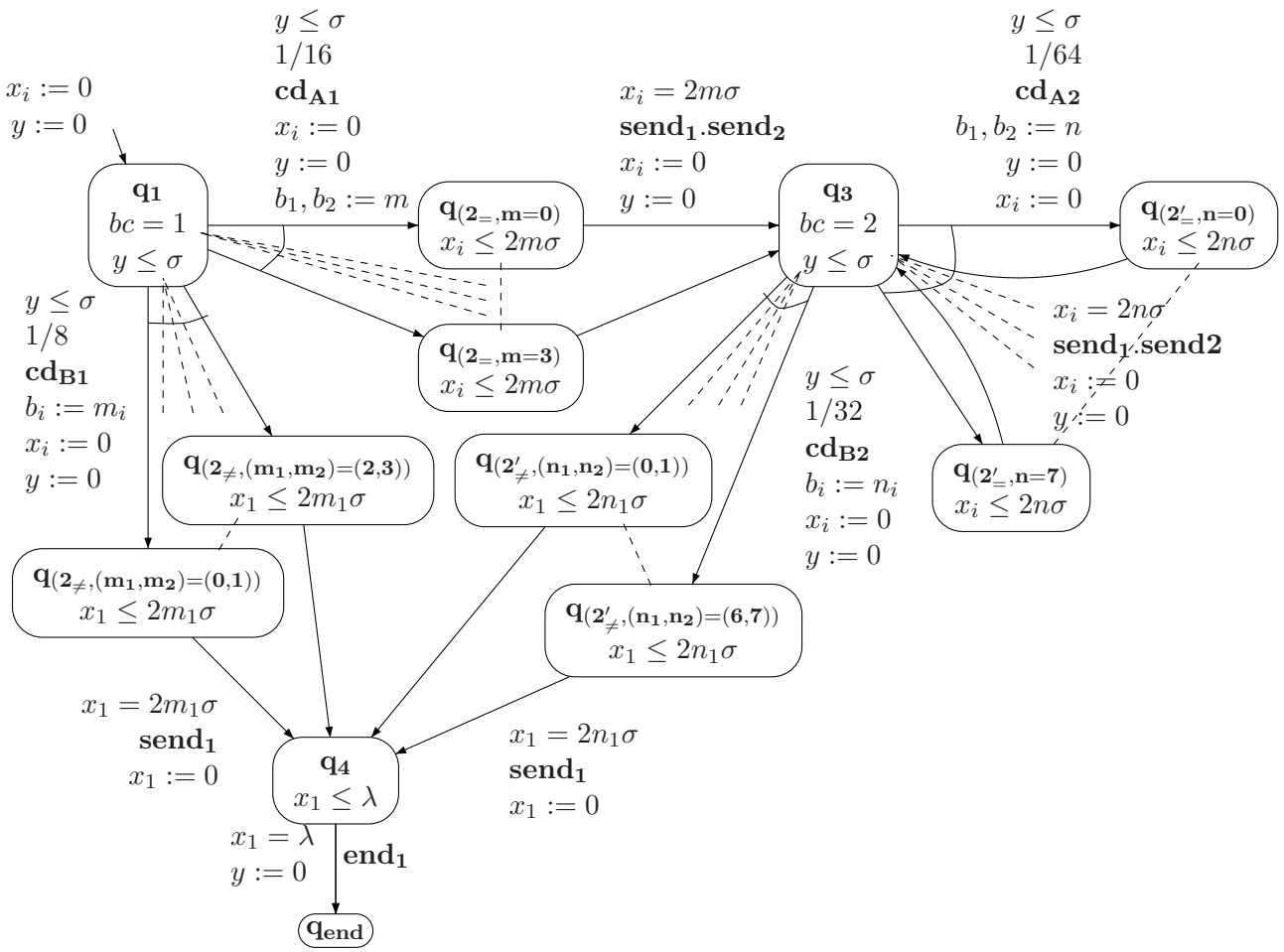

FIG. 3.9 – ATPPSD associé au protocole CSMA/CD

$\psi(q') \in X$. Donc, pour toute valuation ν telle que $\nu \models inv(q)$ et telle qu'il existe $t \in \mathbb{R}^+$ avec $(\nu + t) \models (\psi(q) \leq \phi(q))$, on a $\nu[X := 0] \models inv(q')$ car $\psi(q') \in X$ et $inv(q') = (\psi(q') \leq \phi(q'))$. Ainsi, $\nu \in free_\kappa(q)$.

De façon précise, montrons par exemple que l'état q_3 n'est pas bloquant.

$$free_\kappa(q_3) =\swarrow ((y \leq \kappa(\sigma)) \cap ([x_1, x_2, y := 0] \cap_{m=0}^{7} x_i \leq 2n\kappa(\sigma))) \cap ([x_1, x_2, y := 0] \cap_{n_1=0}^{6} x_1 \leq 2n_1\kappa(\sigma))))$$

Soit ν une valuation telle que $\langle q_3, \nu \rangle$ soit accessible dans $\mathcal{A}_\kappa$. Montrons que $\nu \in free_\kappa(q_3)$.

Toute transition probabiliste e issue de q_3 est de la forme $(q_3, prob(q_3)(q'), y \leq \sigma, \mathcal{X}, q')$. Toute valuation ν qui vérifie $inv(q_3)$, qui est égal à $y \leq \sigma$, vérifie aussi la garde de e, égale à $y \leq \sigma$ (ici t peut être nul). Or l'horloge (x_1 ou x_2) associée à l'état cible q' de la transition e est incluse dans l'ensemble des horloges remises à zéro par cette même transition.
Donc $\nu[x_1, x_2, y := 0] \models inv(q')$. Ainsi, $\nu \in free_\kappa(q_3)$.

Nous constatons enfin que la condition imposée sur les paramètres n'intervient qu'au niveau de l'état q_4 pour achever l'envoi du message par S_1 au travers de la transition end_1.

Nous avons ainsi montré que $\mathcal{A}$ est bien formé pour tout $\kappa \in \Delta$.

$\mathcal{A}$ fortement non zenon.

On doit montrer que dans tout cycle de $\mathcal{A}$, une unité de temps est au moins écoulée comme cela est décrit dans la section 2.5.1.3. Dans l'ATPPSD $\mathcal{A}$, on a 8 cycles qui correspondent aux exécutions de la forme :

$$\langle q_3, \nu_1 \rangle \overset{cd_{A2}}{\rightarrow}_\kappa \langle q_{(2'=,n=i)}, 0 \rangle \overset{send_1 \cdot send_2}{\rightarrow}_\kappa \langle q_3, 0 \rangle,$$

pour tout $i \in \{0, \cdots, 7\}$.

Pour $n \geq 1$, $2n\sigma$ unités de temps sont au moins écoulées dans le cycle correspondant. Pour $n = 0$, le cycle correspondant peut être de durée nulle si la transition probabiliste cd_{A2} est tirée à $y = 0$.

L'automate $\mathcal{A}$ n'est pas donc fortement non zenon.

Dans le cas de l'ATPPD associé à $\mathcal{A}$ de la section 3.4.2 le cas où $n = 0$ ne se pose plus car la garde de la transition probabiliste étiquetée par cd_{A2} est fixée à $y = \sigma$.

3.4.2 L'ATPPD associé à $\mathcal{A}$

Nous considérons l'ATPPSD $\mathcal{A}$ décrit dans la figure 3.9. L'ATPPD associé à $\mathcal{A}$ est obtenu en remplaçant toute garde de la forme $(\psi(q) \leq \phi(q))$ par la garde $(\psi(q) = \phi(q))$.

Dans l'ATPPSD $\mathcal{A}$, nous avons :

– $(\psi(q_1) \leq \phi(q_1))$ avec $\psi(q_1) = y$ et $\phi(q_1) = \sigma$,

– $(\psi(q_3) \leq \phi(q_3))$ avec $\psi(q_3) = y$ et $\phi(q_3) = \sigma$.

Les gardes égales à $(y \leq \sigma)$ dans $\mathcal{A}$ doivent donc être remplacées par la garde $(y = \sigma)$ pour obtenir l'ATPPD associé à $\mathcal{A}$ noté $ATPPD(\mathcal{A})$. L'automate $ATPPD(\mathcal{A})$ est décrit dans la figure 3.10.

$ATPPD(\mathcal{A})$ est bien formé et fortement non zenon car $\mathcal{A}$ est bien formé et fortement non zenon, d'après la proposition 2.23.

3.4.3 Le graphe des macro-steps associé à ATPPD($\mathcal{A}$)

Sans utiliser l'algorithme.

On peut constater que $ATPPD(\mathcal{A})$, qui est une chaîne de Markov avec coûts, est aussi son propre graphe des macro-steps. Ainsi $\widehat{MS}(ATPPD(\mathcal{A}))$ est égal à $ATPPD(\mathcal{A})$.

En effet, ceci est dû au fait que pour tout point à traiter $(X, q) \in Points$ et pour toute transition $e = (q', X, q) \in \mathcal{E}_{(X,q)}$, tout prédécesseur direct de e de l'ensemble $Pred_{directs}(e)$ remet $\psi(q')$ à zéro. La transition e constitue ainsi elle-même un macro-step selon e dont la durée Dur est égale à $\phi(q')$ et dont le poids Wgt est égal à $prob(q')(q)$.

En utilisant l'algorithme.

On adopte une démarche en arrière sur $ATPPD(\mathcal{A})$.
L'ensemble $Points$ est initialisé à $(\mathcal{X}, q_{end})$. L'ensemble des transitions $\mathcal{E}_{(\mathcal{X},q_{end})}$ est égal à $(q_4, \mathcal{X}, q_{end})$. On calcule l'ensemble $Pred_{directs}((q_4, \mathcal{X}, q_{end}))$. Il est égal à l'union de l'ensemble A des transitions $\{(q_{(2_{\neq},(m_1,m_2)=(i,j))}, \{x_1\}, q_4)\}$ avec $i, j \in \{0, \cdots, 3\}$ et $i < j$ et de l'ensemble

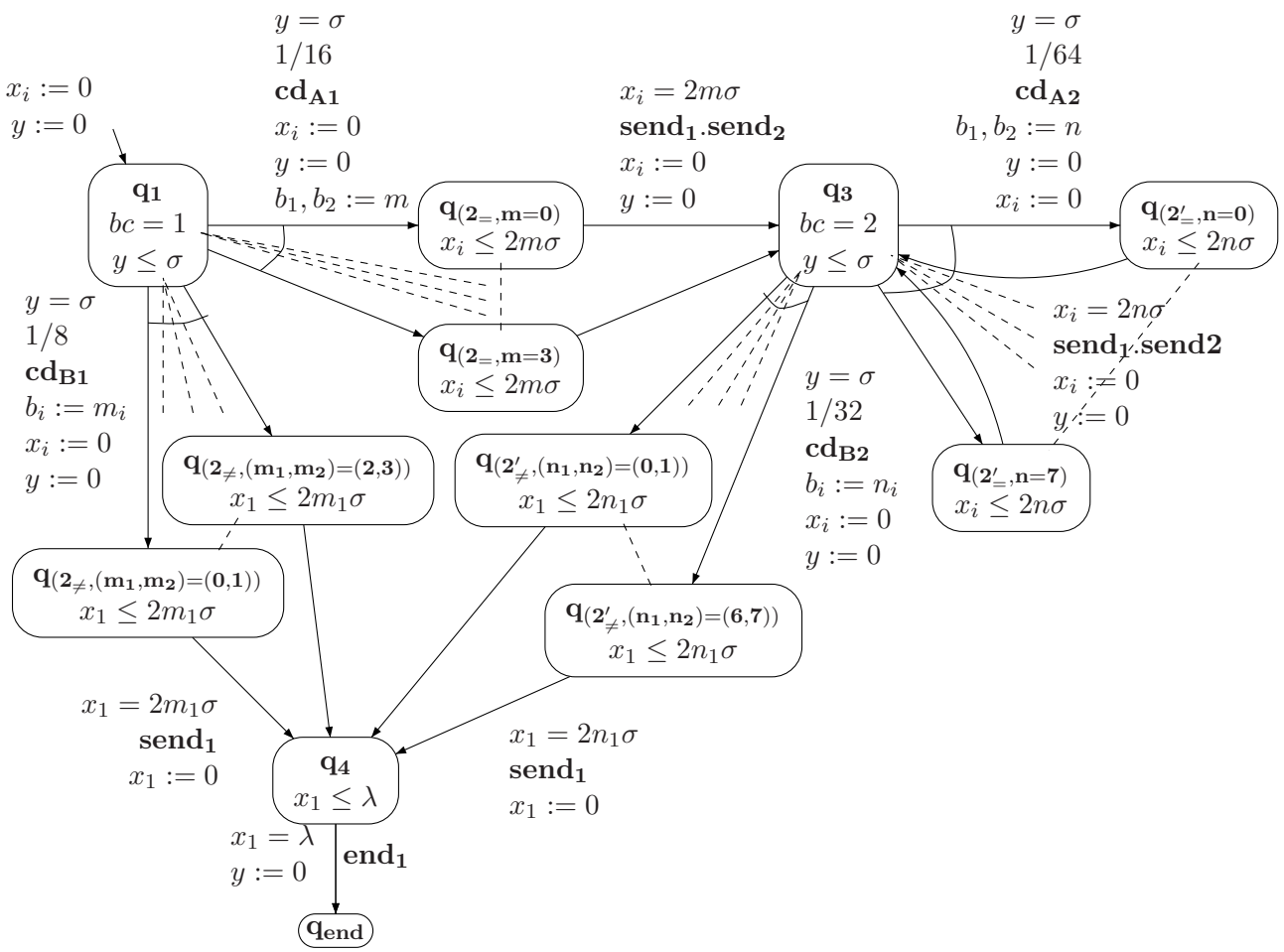

FIG. 3.10 – ATPPD associé au protocole CSMA/CD

A' des transitions $\{(q_{(2'_\neq,(n_1,n_2)=(k,l))}, \{x_1\}, q_4)\}$ avec $k, l \in \{0, \cdots, 7\}$ et $k < l$. Pour tout $e \in Pred_{directs}((q_4, \mathcal{X}, q_{end}))$, e est de la forme $(q', \{x_1\}, q_4)$ où $q' = q_{(2_\neq,(m_1,m_2)=(i,j))}$ ou $q' = q_{(2'_\neq,(n_1,n_2)=(k,l))}$. On a $\psi(q_4) = x_1$ et toute transition $e \in Pred_{directs}((q_4, \mathcal{X}, q_{end}))$ remet l'horloge x_1 à zéro. L'ensemble E de l'algorithme 8 de la section 3.1.3.2 correspond ainsi à l'une des transitions de $Pred_{directs}((q_4, \mathcal{X}, q_{end}))$ car elles ont toutes le même ensemble d'horloges remises à zéro qui est $\{x_1\}$. On crée ainsi le macro-step selon $(q_4, \mathcal{X}, q_{end})$,

$$(\{x_1\}, q_4) \Rightarrow q_{end}.$$

Aucun macro-step ayant même source et même cible n'a déjà été créé. La durée de ce macro-step $Dur((\{x_1\}, q_4) \Rightarrow q_{end})$ est donc égale à $\phi(q_4) = \lambda$ et son poids $Wgt((\{x_1\}, q_4) \Rightarrow q_{end})$ est égal à 1. Le point $(\{x_1\}, q_4)$ est ajouté à l'ensemble $Points$ duquel on supprime le point $(\mathcal{X}, q_{end})$.

On répète la même démarche sur le point $(\{x_1\}, q_4)$. On a $\mathcal{E}_{(\{x_1\},q_4)} = A \cup A'$. Prenons la transition $(q_{(2'_\neq,(n_1,n_2)=(k,l))}, \{x_1\}, q_4) \in A$.

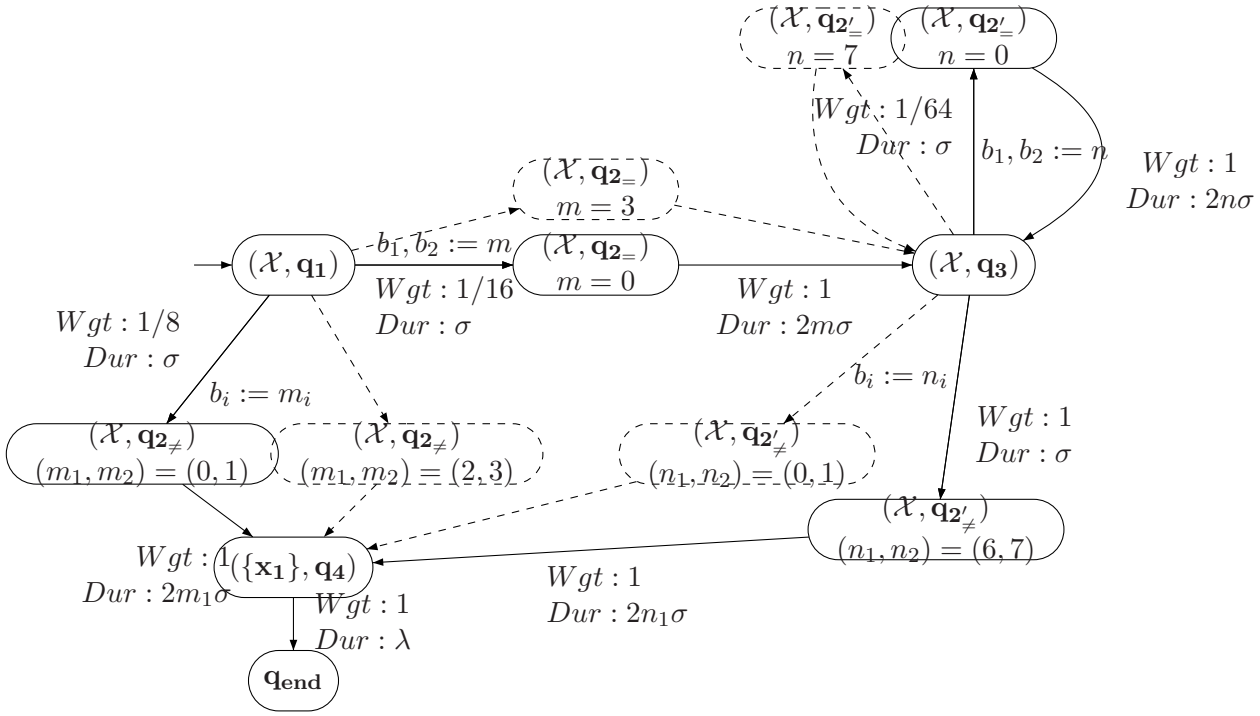

FIG. 3.11 – Graphe des macro-steps associé au protocole CSMA/CD

L'ensemble $Pred_{directs}((q_{(2'_{\neq},(n_1,n_2)=(k,l))}, \{x_1\}, q_4))$ est égal à la seule transition $\{(q_3, \mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))})\}$ qui remet à zéro l'horloge $\psi(q_{(2'_{\neq},(n_1,n_2)=(k,l))})$ qui est égale à x_1. Aucun macro-step de mêmes source et cible n'ayant été déjà créé, on construit un ensemble de nouveaux macro-steps de la forme :

$$(\mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))}) \Rightarrow (\{x_1\}, q_4),$$

pour tout $k \in \{0, \cdots, 6\}$, $l \in \{1, \cdots, 7\}$, $k < l$. Ils sont de durée $2n_1\sigma$ et de poids 1.

On procède de la même manière sur l'ensemble A'. On créé aussi un ensemble de nouveaux macro-steps de la forme :

$$(\mathcal{X}, q_{(2_{\neq},(m_1,m_2)=(i,j))}) \Rightarrow (\{x_1\}, q_4),$$

pour tout $i \in \{0, \cdots, 2\}$, $j \in \{1, \cdots, 3\}$, $i < j$. Ils sont de durée $2m_1\sigma$ et de poids 1.

On ajoute tous les points de la forme $(\mathcal{X}, q_{(2_{\neq},(m_1,m_2)=(i,j))})$ et $(\mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))})$ à l'ensemble $Points$ et on en supprime le point $(\{x_1\}, q_4)$.

Prenons le point $(\mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))})$ de l'ensemble $Points$. L'ensemble $\mathcal{E}_{(\mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))})}$ est égal à la transition $(q_3, \mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))})$.
L'ensemble $Pred_{directs}((q_3, \mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))}))$ est égal à l'ensemble B' des transitions $\{(q_{(2'_{=},(n=k))}, \mathcal{X}, q_3)\}$ avec $k \in \{0, \cdots, 7\}$ et à l'ensemble B des transitions $\{(q_{(2_{=},(m=i))}, \mathcal{X}, q_3)\}$ avec $i \in \{0, \cdots, 3\}$.
Les transitions de l'ensemble $Pred_{directs}((q_3, \mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))}))$ ont le même ensemble d'horloges remises à zéro et remettent $\psi(q_3) = y$ à zéro. On créé un premier ensemble de nouveaux macro-steps de la forme :

$$(\mathcal{X}, q_3) \Rightarrow (\mathcal{X}, q_{(2'_{\neq},(n_1,n_2)=(k,l))}),$$

pour tout $k \in \{0, \cdots, 6\}$, $l \in \{1, \cdots, 7\}$, $k < l$. Ils sont de durée σ et de poids $1/32$.

On construit un deuxième ensemble de nouveaux macro-steps de la forme :

$$(\mathcal{X}, q_1) \Rightarrow (\mathcal{X}, q_{(2_{\neq},(m_1,m_2)=(i,j))}),$$

pour tout $i \in \{0, \cdots, 2\}$, $j \in \{1, \cdots, 3\}$, $i < j$. Ils sont de durée σ et de poids $1/8$.

Prenons le point $(\mathcal{X}, q_3)$. L'ensemble $\mathcal{E}_{(\mathcal{X},q_3)}$ est égal à $B \cup B'$. Soit une transition de la forme $(q_{(2'_{=},(n=k))}, \mathcal{X}, q_3) \in B'$. L'ensemble $Pred_{directs}((q_{(2'_{=},(n=k))}, \mathcal{X}, q_3))$ est égal à la transition $\{(q_3, \mathcal{X}, q_{(2'_{=},(n=k))})\}$ qui remet l'horloge $\psi(q_{(2'_{=},(n=k))})$ à zéro. On créé ainsi un nouvel ensemble de macro-steps de la forme :

$$(\mathcal{X}, q_{(2'_{=},(n=k))}) \Rightarrow (\mathcal{X}, q_3),$$

pour tout $k \in \{0, \cdots, 7\}$. Ils sont de durée $2n\sigma$ et de poids $1/64$.

De la même manière on crée un nouvel ensemble de macro-steps de la forme :

$$(\mathcal{X}, q_{(2'_{=},(m=i))}) \Rightarrow (\mathcal{X}, q_3),$$

pour tout $i \in \{0, \cdots, 3\}$. Ils sont de durée $2m\sigma$ et de poids 1.

Prenons maintenant un point de la forme $(\mathcal{X}, q_{(2'_{=},(n=k))})$. L'ensemble $\mathcal{E}_{(\mathcal{X},q_{(2'_{=},(n=k))})}$ est égal à la transition $\{(q_3, \mathcal{X}, q_{(2'_{=},(n=k))})\}$.
L'ensemble $Pred_{directs}((q_3, \mathcal{X}, q_{(2'_{=},(n=k))}))$ est égal à l'ensemble B' dont chaque transition remet à zéro l'horloge $\psi(q_3) = y$. On crée ainsi un nouvel ensemble de macro-steps de la forme :

$$(\mathcal{X}, q_3) \Rightarrow (\mathcal{X}, q_{(2'_{=},(n=k))}),$$

pour tout $k \in \{0, \cdots, 7\}$. Ils sont de durée σ et de poids $1/64$.

De la même manière on crée le nouvel ensemble de macro-steps de la forme :

$$(\mathcal{X}, q_1) \Rightarrow (\mathcal{X}, q_{(2_{=},(m=i))}),$$

pour tout $i \in \{0, \cdots, 3\}$. Ils sont de durée σ et de poids $1/16$.

On ne rajoute pas le point $(\mathcal{X}, q_1)$ à l'ensemble $Points$ car q_1 est l'état initial de l'automate $ATPPD(\mathcal{A})$.

La construction du graphe de macros-steps se termine à cette étape car l'ensemble $Points$ est vide.

3.4.4 Calcul du pire temps moyen de convergence dans $\mathcal{A}$

3.4.4.1 Unique état final absorbant

On souhaite calculer le pire temps moyen de convergence (i.e temps moyen de convergence maximal) dans $\mathcal{A}$ pour atteindre un état final qui traduit l'envoi d'un message complet avec succès par l'un des deux émetteurs. Cet état final sera l'état $q_{end} = Init_c \cdot Done_1 \cdot Wait_2$ (ou par symétrie $q_{end} = Init_c \cdot Done_2 \cdot Wait_1$).

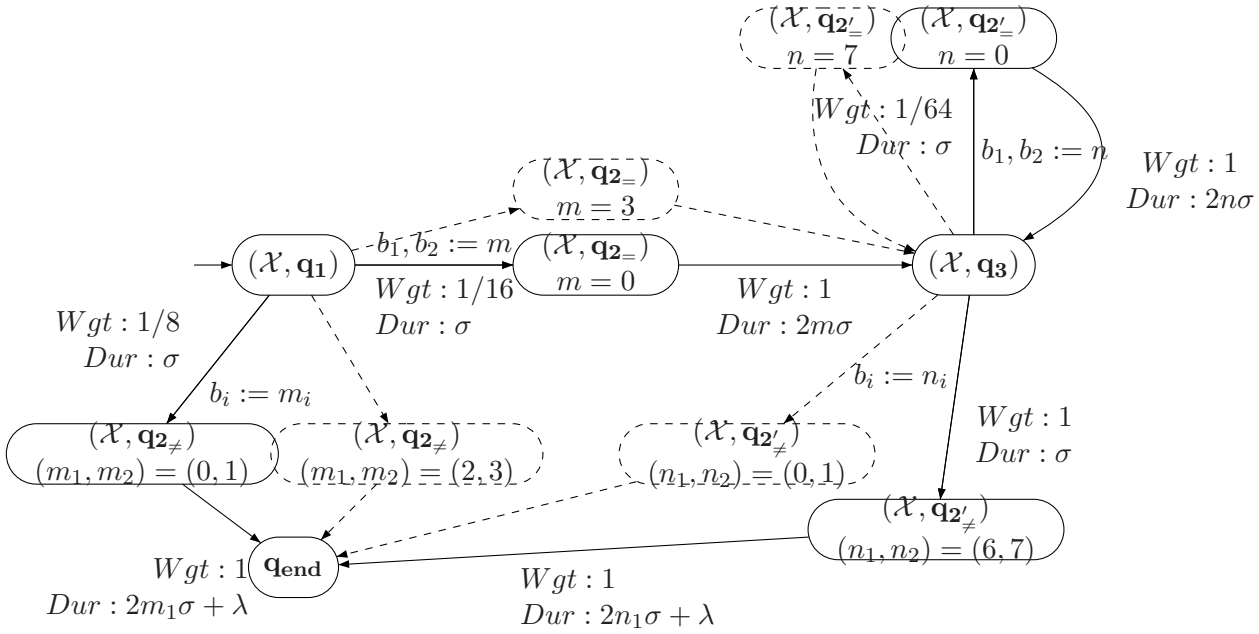

FIG. 3.12 – Graphe des macro-steps utilisé pour le calcul

L'état q_{end} est aussi absorbant car il constitue la seule classe récurrente $\{q_{end}\}$ (voir la section 2.2) de l'ATPPSD $\mathcal{A}$ décrit dans la figure 3.9. En effet, le singleton $\{q_{end}\}$ est le seul ensemble de l'ATPPSD $\mathcal{A}$ qu'on ne peut plus quitter une fois qu'on y entre.

3.4.4.2 Résolution

Le calcul du temps moyen de convergence $MExpAbs(ATPPD(\mathcal{A}))$ vers l'état q_{end} est appliqué sur le graphe des macro-steps décrit dans la figure 3.12. Il est obtenu à partir du graphe des macro-steps initial décrit dans la figure 3.11 en fusionnant les états q_4 et q_{end}.
Les macro-steps de la forme $(\mathcal{X}, q_{(2_\neq,(m_1,m_2)=(i,j))}) \Rightarrow q_4$ et
$(\mathcal{X}, q_{(2'_\neq,(n_1,n_2)=(k,l))}) \Rightarrow q_4$ de durées respectives $2m_1\sigma$ et $2n_1\sigma$ sont remplacés par les macro-steps $(\mathcal{X}, q_{(2_\neq,(m_1,m_2)=(i,j))}) \Rightarrow q_{end}$ et $(\mathcal{X}, q_{(2_\neq,(m_1,m_2)=(i,j))}) \Rightarrow q_{end}$ de durées respectives $2m_1\sigma + \lambda$ et $2n_1\sigma + \lambda$.

L'ensemble $\mathcal{V}$ est composé des 49 points constituant les états du graphe des macro-steps décrit dans la figure 3.12. L'ensemble $\mathcal{V}'$ est égal à $\mathcal{V} \setminus q_{end}$.
La matrice carrée $\mathbf{M}$ est définie sur l'ensemble $\mathcal{V}'$. Elle est donc de dimension 48.

On a vu dans la section 3.4.3 que le graphe des macro-steps associé à $ATPPD(\mathcal{A})$ est l'automate $ATPPD(\mathcal{A})$ lui-même, qui est une chaîne de Markov à coûts. On peut ainsi conclure sans calcul que le vecteur W est égal au vecteur unité.
Le calcul est effectué à l'aide de l'outil Maple adapté au calcul matriciel avec paramètres.

```
> restart:
```

```
> with(linalg):
```

On définit ci-dessous la matrice $\mathbf{M}$ de dimension 48. On associe un entier $i \in \{1, \cdots, 48\}$ à chaque état du graphe du macro-steps :

- L'état q_1 est représenté par 1.
- Les entiers de 2 à 5 représentent dans l'ordre les états $q_{(2_=, m=j)}$, $j \in \{0, ..., 3\}$.
- $i = 6$ représente l'état q_3.
- Les entiers de 7 à 14 représentent dans l'ordre les états $q_{(2'_=, n=j)}$, $j \in \{0, ..., 7\}$.
- Les entiers de 15 à 42 représentent dans l'ordre les états $q_{(2'_{\neq}, (n_1,n_2)=(k,l))}$, $j \in \{0, ..., 6\}$, $k \in \{1, ..., 7\}$ et $k < l$.
- Les entiers de 43 à 48 représentent dans l'ordre les étas $q_{(2_{\neq}, (m_1,m_2)=(s,t))}$, $s \in \{0, ..., 2\}$, $t \in \{1, ..., 3\}$ et $s < t$.

Le code en Maple relatif à cette description est le suivant :

```
> M:=matrix(48,48,0):

> for i from 2 to 5 do M[1,i]:=1/16; M[i,6]:=1 end do:

> for i from 7 to 14 do M[6,i]:=1/64; M[i,6]:=1 end do:

> for i from 15 to 42 do M[6,i]:=1/32 end do:

> for i from 43 to 48 do M[1,i]:= 1/8 end do:

> evalm(M):
```

Ceci est donc la matrice de transitions M.

On calcule maintenant la matrice $I - \mathbf{M}$ de dimension 48. On la note N. On vérifie que son déterminant est bien différent de zéro.

Le code Maple est le suivant :

```
> N:=matrix(48,48):

> for i from 1 to 48 do for j from 1 to 48 do if (i=j) then N[i,i]:=-M[i,i]+1
else N[i,j]:=-M[i,j] end if: end do: end do:

>

> evalm(N): det(N);
```

$$\frac{7}{8}$$

Z est l'inverse de la matrice $I - \mathbf{M}$.

Le code Maple est le suivant :

```
> Z:=inverse(N): evalm(Z):
```

On définit le vecteur B de dimension 48 égal à $(Wgt((X,q) \Rightarrow q_{end}))_{(X,q)\in\mathcal{V}'}$.

Le code Maple est le suivant :

```
> B:=vector(48,0):
> for i from 15 to 48 do B[i]:=1 end do:
>
```

On vérifie ici que le graphe des macro-steps associé à $ATPPD(\mathcal{A})$ est bien une chaîne de Markov avec coûts car $W = 1$.

Le code Maple est le suivant :

```
> W := multiply(Z,B): evalm(W);
```

$[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1$
$, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$

On calcule ci-dessous le vecteur C égal à $Q + R$ avec $Q = AW$.

$$C(X,q) = \sum_{(Y,q')\in\mathcal{V}'} Dur((X,q) \Rightarrow (Y,q'))\mathbf{M}((X,q),(Y,q'))w(Y,q')+Dur((X,q) \Rightarrow q_{end})Wgt((X,q) \Rightarrow q_{end}).$$

Donc

$$R(X,q) = Dur((X,q) \Rightarrow q_{end})Wgt((X,q) \Rightarrow q_{end}),$$

$W = 1$ et $A((X,q),(Y,q')) = Dur((X,q) \Rightarrow (Y,q'))\mathbf{M}((X,q),(Y,q'))$.

Le code Maple est le suivant :

Calcul de R :

```
> R:=vector(48,0):
> for i from 15 to 21 do R[i]:=lambda end do:
> for i from 22 to 27 do R[i]:=2*sigma+lambda end do:
> for i from 28 to 32 do R[i]:=4*sigma+lambda end do:
> for i from 33 to 36 do R[i]:=6*sigma+lambda end do:
> for i from 37 to 39 do R[i]:=8*sigma+lambda end do:
> for i from 40 to 41 do R[i]:=10*sigma+lambda end do:
> R[42]:=12*sigma+lambda:
> for i from 43 to 45 do R[i]:=lambda end do:
> for i from 46 to 47 do R[i]:=2*sigma+lambda end do:
> R[48]:=4*sigma+lambda:
> evalm(R);
```

$[0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ \lambda,\ \lambda,\ \lambda,\ \lambda,\ \lambda,\ \lambda,\ \lambda,\ 2\,\sigma+\lambda,\ 2\,\sigma+\lambda,\ 2\,\sigma+$
$\lambda,\ 2\,\sigma+\lambda,$
$2\,\sigma+\lambda,\ 2\,\sigma+\lambda,\ 4\,\sigma+\lambda,\ 4\,\sigma+\lambda,\ 4\,\sigma+\lambda,\ 4\,\sigma+\lambda,\ 4\,\sigma+\lambda,\ 6\,\sigma+\lambda,\ 6\,\sigma+\lambda,$
$6\,\sigma+\lambda,\ 6\,\sigma+\lambda,\ 8\,\sigma+\lambda,\ 8\,\sigma+\lambda,\ 8\,\sigma+\lambda,\ 10\,\sigma+\lambda,\ 10\,\sigma+\lambda,\ 12\,\sigma+\lambda,\ \lambda,\ \lambda,\ \lambda,$
$2\,\sigma+\lambda,\ 2\,\sigma+\lambda,\ 4\,\sigma+\lambda]$

>

>

Calcul de A :

```
> A:=matrix(48,48,0): for i from 2 to 5 do A[1,i]:=sigma/16; end do:
> for i from 2 to 5 do A[i,6]:=(2*(i-2))*sigma end do:
> for i from 15 to 42 do A[6,i]:=sigma/32: end do:
>
> for i from 7 to 14 do A[6,i]:=sigma/64; end do:
> for i from 7 to 14 do A[i,6]:=(2*(i-7))*sigma; end do:
>
> for i from 43 to 48 do A[1,i]:=sigma/8 end do:
> evalm(A):
```

Calcul de $Q = AW$:

```
> Q:=multiply(A,W):
```

Calcul de C qui est égal à $Q + R$:

```
> C:=matadd(Q,R):
```

Enfin, le vecteur $T = (MExpAbs(X, q))_{(X,q)\in\mathcal{V}'}$ est égal à $(I - \mathbf{M})^{-1}C$.

Le code Maple correspondant est le suivant :

```
> T:=multiply(Z,C);
```

$$
\begin{aligned}
T :=\ & [\lambda + \tfrac{30}{7}\,\sigma, \lambda + \tfrac{43}{7}\,\sigma, \lambda \\
& + \tfrac{57}{7}\,\sigma, \lambda + \tfrac{71}{7}\,\sigma, \lambda + \tfrac{85}{7}\,\sigma \\
& , \lambda + \tfrac{43}{7}\,\sigma, \lambda + \tfrac{43}{7}\,\sigma, \lambda + \tfrac{57}{7}\,\sigma \\
& , \lambda + \tfrac{71}{7}\,\sigma, \lambda + \tfrac{85}{7}\,\sigma, \lambda + \tfrac{99}{7}\,\sigma \\
& , \lambda + \tfrac{113}{7}\,\sigma, \lambda + \tfrac{127}{7}\,\sigma, \lambda + \tfrac{141}{7}\,\sigma \\
& , \lambda, \lambda, \lambda, \lambda, \lambda, \lambda, \lambda, \lambda, 2\,\sigma + \lambda, 2\,\sigma + \lambda, 2\,\sigma + \lambda, 2\,\sigma + \lambda, 2\,\sigma + \lambda, 2\,\sigma + \lambda, 4\,\sigma + \lambda, 4\,\sigma + \lambda, 4\,\sigma + \\
& \lambda, 4\,\sigma + \lambda, 4\,\sigma + \lambda, 6\,\sigma + \lambda, 6\,\sigma + \lambda, 6\,\sigma + \lambda \\
& , 6\,\sigma + \lambda, 8\,\sigma + \lambda, 8\,\sigma + \lambda, 8\,\sigma + \lambda, 10\,\sigma + \lambda, 10\,\sigma + \lambda, 12\,\sigma + \lambda, \lambda, \lambda, \lambda, 2\,\sigma + \lambda, 2\,\sigma + \lambda, 4\,\sigma + \lambda]
\end{aligned}
$$

>

3.4.4.3 Résultat

Nous avons obtenu le vecteur T dans la section 3.4.4.2. Le temps moyen de convergence vers l'état q_{end} dans $\widehat{MS}(ATPPD(\mathcal{A}))$ en partant de l'état $(\mathcal{X}, q_1)$ est égal à la première composante du vecteur T, soit $T[1]$. On conclut donc que

$$MExpAbs(ATPPD(\mathcal{A})) = \frac{30}{7}\sigma + \lambda.$$

$MExpAbs(ATPPD(\mathcal{A}))$ est donc une combinaison linéaire des paramètres du protocole σ et λ.
Soit κ une valuation paramétrique vérifiant la condition $\lambda \geq 2\sigma$ telle que $\kappa(\sigma) = 26$ et $\kappa(\lambda) = 808$. Par la propostion 3.2, $\kappa(MExpAbs(ATPPD(\mathcal{A})))$ est le temps moyen de convergence $ExpAbs(ATPPD(\mathcal{A}), \kappa)$ pour atteindre q_{end} en partant de q_1 dans l'automate $ATPPD(\mathcal{A})$. On en déduit le résultat suivant :
$ExpAbs(ATPPD(\mathcal{A}), \kappa) = \frac{30}{7} \times 26 + 808 = 919.42$ [1].

Le pire temps moyen de convergence (i.e temps moyen de convergence maximal) pour atteindre q_{end} en partant de q_1 dans $\mathcal{A}$ est égal à $ExpAbs(ATPPD(\mathcal{A}), \kappa)$. Ainsi, le pire temps moyen nécessaire pour que l'un des émetteurs envoie un message complet avec succès après qu'une première collision n'ait eu lieu est égal à $ExpAbs(ATPPD(\mathcal{A}), \kappa)$.

Conclusion

Nous avons présenté dans cette thèse deux principaux points : une sous classe d'automates temporisés probabilistes et un calcul paramétré du temps moyen de convergence d'un automate de cette sous classe vers un état final et absorbant.

La sous classe d'automates temporisés probabilistes appelée Automates Temporisés Probabilistes Paramétrés Déterminés (ATPPD) se définit principalement par l'absence totale de non déterminisme, qu'il soit d'action ou de temps. Le temps d'attente en chaque état est déterminé par la valuation des horloges à l'entrée de l'automate car les gardes sont des égalités de la forme $x = a$. De plus, une seule distribution est associée à chaque état. La troisième caractéristique de cette sous classe réside dans la forme des données temporelles au niveau des gardes ou des invariants car elle peuvent être représentées par des paramètres ou des entiers naturels. Grâce aux paramètres, on représente un ensemble d'automates temporisés probabilistes par un seul automate car chaque paramètre peut prendre plusieurs valeurs dans l'ensemble des entiers naturels. Le calcul que nous proposons est paramétré et permet donc de donner le temps moyen de convergence d'un ensemble d'automates si l'on donne des valeurs aux paramètres.

Afin de calculer le temps moyen de convergence vers un état spécifique pour cette classe d'automates, on fait appel à trois hypothèses : l'automate doit être bien formé, fortement non zenon et doit posséder un unique état final et absorbant. La propriété d'automate bien formé est vérifiée sur le graphe des zones associé à l'automate. On s'assure par des algorithmes appliqués sur le graphe des zones qu'il n'existe pas d'états bloquants. Ceci conduit à déduire que de tout état de l'automate on atteint l'état final. Par ailleurs, l'automate doit être fortement non zenon pour éviter d'avoir des cycles de durée nulle. Cette propriété peut être contournée en déterminant les cas fortement zenon et en les isolant dans la suite lors du calcul. Enfin, l'automate doit avoir un unique état final et absorbant pour calculer le temps moyen de convergence vers cet état spécifique. Cet état doit donc constituer l'unique classe récurrente de l'automate.

On détermine un ensemble de valeurs possibles pour chaque paramètre de l'automate pour vérifier ces propriétés, ce qui peut mener à faire certaines restrictions sur le choix de ces valeurs.

D'autre part, cette sous classe d'automate est motivée par le fait que le pire temps moyen de convergence est capturé par un ATPPD dans certains cas de systèmes distribués modélisés par des automates temporisés probabilistes comme par exemple les protocoles BRP ou CSMA/CD, En effet, si certaines réductions sont faites sur l'automate initial en vue de calculer le pire temps moyen de convergence, elles peuvent permettre de supprimer quelques cas de non déterminisme d'actions au niveau des distributions possibles en

chaque état. De plus, le temps écoulé en chaque état est fixé par un adversaire ou scheduler qui y reste le plus longtemps possible. Certaines contraintes imposées aux paramètres pour répondre aux spécifications physiques du système peuvent aussi faire disparaître des cas de non déterminisme dans le choix de la distribution pour certains états.

Un calcul paramétré permet d'obtenir le temps moyen de convergence dans un ATPPD. A partir de l'ATPPD, on construit tout d'abord un graphe appelé graphe des macro-steps. C'est un graphe semblable aux automates à coûts positifs de par la forme sauf que la construction fait que la somme des probabilités des chemins issus d'un état donné n'est pas toujours égale à 1. On applique ensuite la méthode de calcul du temps moyen développé par [6] sur les automates à coûts positifs au graphe des macro-steps. On résout deux systèmes d'équations linéaires qui permettent d'obtenir une combinaison linéaire des paramètres de l'ATPPD. La construction du graphe des macros-steps se fait en temps polynomial, de même que la résolution des systèmes linéaires contrairement à la méthode proposée dans [24] qui est exponentielle car elle discrétise le temps. La méthode que nous proposons se caractérise ainsi par son aspect paramétré et par un calcul dont la taille ne dépend pas de la plus grande constante de l'automate.

On démontre d'autre part que le temps moyen de convergence dans un ATPPD est égal au temps moyen de convergence dans son graphe des macro-steps. Cette égalité est obtenue en montrant l'existence d'une correspondance entre les chemins de l'ATPPD et ceux de son graphe des macro-steps.

Le résutat obtenu étant paramétré, on attribue à chaque paramètre une valeur entière pour avoir le temps moyen de convergence dans l'ATPPD. Ce temps correspond finalement au pire temps moyen de convergence de l'automate initial dont découle l'ATPPD.

Bibliographie

[1] Prism web site. http ://www.prismmodelchecker.org/.

[2] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2) :183–235, 1994.

[3] R. Alur, T. A. Henzinger, and M. Y. Vardi. Parametric real-time reasoning. In *Proc. STOC'93*, pages 592–601. ACM, 1993.

[4] C. Baier, M. Größer, and F. Ciesinski. Partial order reduction for probabilistic systems. In *Proc. QEST'04*, pages 230–239. IEEE Computer Society, 2004.

[5] B. Bérard, V. Diekert, P. Gastin, and A. Petit. Characterization of the expressive power of silent transitions in timed automata. *Fundamenta Informaticae*, 36(2) :145–182, November 1998.

[6] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation : Numerical Methods*. Prentice-Hall, 1989.

[7] D.P. Bertsekas and J.N. Tsitsiklis. An Analysis of Stochastic Shortest Path Problems. *Mathematics of Operations Research 16 :3*, pages 580–595, 1991.

[8] Dirk Beyer. Improvements in bdd-based reachability analysis of timed automata. In *FME*, volume 2021 of *LNCS*, pages 318–343. Springer, 2001.

[9] Dirk Beyer and Andreas Noack. Efficient verification of timed automata using bdds. In *Proceedings of the 6th International ERCIM Workshop on Formal Methods for Industrial Critical Systems (FMICS 2001*, pages 95–113, 2001.

[10] Patricia Bouyer. Forward analysis of updatable timed automata. *Formal Methods in System Design*, 24(3) :281–320, May 2004.

[11] Patricia Bouyer. Weighted timed automata : Model-checking and games. *Electr. Notes Theor. Comput. Sci.*, 158 :3–17, 2006.

[12] P. Brémaud. *Markov Chains*. Springer, 1999.

[13] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Second Edition*. The MIT Press and McGraw-Hill Book Company, 2001.

[14] P. D'Argenio, B. Jeannet, H. Jensen, and K. Larsen. Reachability Analysis of Probabilistic Systems by Successive Refinements. In *PAPM-PROBMIV'01, LNCS 2165, Springer*, pages 39–56, 2001.

[15] P. D'Argenio, J.-P. Katoen, T.C. Ruys, and J. Tretmans. The bounded retransmission protocol must be on time ! In *Proc. TACAS'97*, volume 1217 of *LNCS*, pages 416–431. Springer, 1997.

[16] L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Stanford University edition, 1997.

[17] L. de Alfaro. Computing Minimum and Maximum Reachability Times in Probabilistic Systems. In *CONCUR 99, LNCS 1664, Springer*, pages 66–81, 1999.

[18] Laurent Doyen. Robust parametric reachability for timed automata. *Inf. Process. Lett.*, 102(5) :208–213, 2007.

[19] T. Henzinger. *The Temporal Specification and Verification of Real-time Systems*. PhD Thesis. Stanford University, 1991.

[20] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. A User Guide to HYTECH. In *TACAS'95*, volume 1019 of *LNCS*, pages 41–71. Springer, 1995.

[21] Thomas A. Henzinger, Zohar Manna, and Amir Pnueli. What good are digital clocks ? In *ICALP*, volume 623 of *LNCS*, pages 545–558. Springer, 1992.

[22] R. A. Howard. *Dynamic Probabilistic Systems*. John Wiley and Sons, 1971.

[23] J. G. Kemeny, J. L. Snell, and A. W Knapp. *Denumerable Markov Chains*. Graduate Texts in Mathematics. Springer, 2nd edition, 1976.

[24] M. Kwiatkowska, G. Norman, D. Parker, and J. Sproston. Performance Analysis of Probabilistic Timed Automata using Digital Clocks. *Formal Methods in System Design*, 29(1) :33–78, 2006.

[25] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic verification of real-time systems with discrete probability distributions. *Theoretical Computer Science*, 282(1) :101–150, 2002.

[26] M. Kwiatkowska, G. Norman, and J. Sproston. Probabilistic model checking of deadline properties in the IEEE 1394 FireWire root contention protocol. *Formal Aspects of Computing*, 14(3) :295–318, 2003.

[27] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang. Symbolic model checking for probabilistic timed automata. In Y. Lakhnech and S. Yovine, editors, *Proc. Joint Conference on Formal Modelling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant Systems (FORMATS/FTRTFT'04)*, volume 3253 of *LNCS*, pages 293–308. Springer, 2004.

[28] M. Kwiatkowska, G. Norman, J. Sproston, and F. Wang. Symbolic model checking for probabilistic timed automata. *Information and Computation*, 205(7) :1027–1077, 2007.

[29] F. Laroussinie and J. Sproston. State explosion in almost-sure probabilistic reachability. *Information Processing Letters*, 102(6) :236–241, 2007.

[30] M. L. Puterman. *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. John Wiley and Sons, 1994.

[31] R. Segala and N. Lynch. Probabilistic Simulations for Probabilistic Processes. *Nordic Journal of Computing 2 :2*, pages 250–273, 1995.

[32] Roberto Segala. Compositional verification of randomized distributed algorithms. In *COMPOS*, volume 1536 of *LNCS*, pages 515–540. Springer, 1997.

[33] D. P. L. Simons and M. Stoelinga. Mechanical verification of the IEEE 1394a root contention protocol using Uppaal2k. *Software Tools for Technology Transfer*, 3(4) :469–485, 2001.

[34] Stavros Tripakis. Verifying progress in timed systems. In *ARTS*, volume 1601 of *LNCS*, pages 299–314. Springer, 1999.

[35] Douglas B. West. *Introduction to Graph Theory.* Prentice Hall 1996, 2001.