EcoMata : Un logiciel d'aide à la décision pour améliorer la gestion des écosystèmes

Yulong Zhao, Christine Largouët, Marie-Odile Cordier

EcoMata : Un logiciel d'aide à la décision pour améliorer la gestion des écosystèmes

Yulong Zhao[1,3] — Christine Largouët[1,2] — Marie-Odile Cordier[1,3]

[1] *IRISA, Campus universitaire de Beaulieu, 35042 Rennes, France*

[2] *AGROCAMPUS OUEST, 65 rue de St-Brieuc, 35042 Rennes, France*

[3] *Université de Rennes 1, Campus universitaire de Beaulieu 35042 Rennes, France*

contact : yulong.zhao@irisa.fr, largouet@agrocampus-ouest.fr, cordier@irisa.fr

RÉSUMÉ. *Les modèles de simulation sont reconnus, en particulier dans le domaine de l'écologie, comme un moyen de mieux comprendre les liens complexes existant entre les actions humaines, le contexte environnemental et les réponses de l'écosystème à ces perturbations. L'approche que nous proposons s'appuie sur une modélisation qualitative de type systèmes à événements discrets . Nous y associons un langage de requête de haut niveau qui permet à l'utilisateur d'exprimer le scénario qu'il veut explorer. Cette approche est implémentée dans le logiciel EcoMata. Ce logiciel permet une construction interactive d'un réseau d'automates, son interrogation automatisée à l'aide des patrons de scénarios et une restitution des résultats de l'interrogation sous une forme conviviale. Nous illustrons notre approche dans le domaine halieutique.*

ABSTRACT. *In the domain of ecology, simulation models are often used to help people understand the complex relationships existing between human actions and nature as well as to anticipate the reactions of the ecosystem to human disturbances. Our approach associates qualitative modeling by a discrete-event system, and a high-level query language that lets users formulate evolution scenarios in the form of database queries. A software named EcoMata has been implemented to bring the approach to real users.*

MOTS-CLÉS : *modélisation qualitative, scénarios prédictifs, logiciel d'aide à la décision, modèle à événements discrets, langage de requête, automates temporisés, model-checking*

KEYWORDS: *qualitative modeling, predictive scenario, decision support system, discrete event system, query language, timed-automata, model-checking*

1. Introduction

L'utilité des modèles de simulation est tout à fait reconnue, en particulier dans le domaine de l'écologie où il est essentiel de mieux comprendre les liens complexes existant entre les actions humaines, le contexte environnemental (climat par exemple) et les réponses de l'écosystème à ces perturbations. Les modélisations d'écosystèmes sont souvent des modélisations à base d'équations mathématiques comme le remarque (Rykiel, 1989). Il est cependant reconnu que ces modèles numériques sont souvent complexes et en conséquence peu adaptés dans un contexte d'aide à la décision, où la modélisation et le langage de requête doivent être faciles à appréhender par l'utilisateur qui n'est, en général, ni un expert ni un chercheur, mais un décideur, tel que le gestionnaire des pêches dans notre exemple. Les outils d'aide à la décision sont d'autant plus utiles que les temps de réponse permettent une utilisation interactive des différents scénarios de tests. Ceci est en particulier vrai pour les systèmes dynamiques dont l'évolution dans le temps est complexe comme dans le cas du contrôle de population. La dynamique d'évolution de chaque population (dans notre cas les espèces de poisson) est relativement facile à exprimer de manière qualitative. En revanche, prédire ce qui va se passer, suite à l'application par exemple d'une politique de pêche, est difficile, en raison de l'imbrication des interactions, surtout si l'on veut prendre en compte le contexte environnemental tel que le climat ou les catastrophes (ouragan, inondation. . .).

La simulation qualitative a été utilisée de manière satisfaisante dans de nombreux domaines liés à l'environnement tels que la physiologie végétale (Rickel *et al.*, 1997; Veresi *et al.*, 2010), l'écologie terrestre (Salles *et al.*, 2006) et aquatique (Tulllos *et al.*, 2006; Guerrin *et al.*, 2001), et la pollution de l'eau (Beaujouan *et al.*, 2001; Cordier *et al.*, 2005). Cependant dans un contexte d'aide à la décision, il reste difficile à l'utilisateur de fournir les bonnes entrées au modèle, les données n'étant pas toujours connues ou disponibles, et d'analyser les résultats de simulation, souvent trop abondants. Notre approche s'appuie sur une modélisation qualitative de type systèmes à événements discrets, qui semble bien adaptée pour représenter la dynamique des écosystèmes. Les états représentent les états qualitatifs du système (états stables et états transitoires) ; les transitions représentent les événements provoquant les changements d'états, correspondant à des actions contrôlables, telles que des actions humaines, ou incontrôlables telles que des événements climatiques. Des horloges peuvent être associées aux états et aux transitions, afin de représenter de manière explicite les contraintes temporelles contrôlant la dynamique de l'évolution du système.

S'appuyant sur une modélisation qualitative que l'on peut considérer comme une représentation abstraite d'un système continu, des travaux ont déjà montré l'intérêt d'employer des techniques de model-checking (Clarke *et al.*, 2002; Berard *et al.*, 2001) pour vérifier des propriétés du système. La modélisation qualitative peut, en effet, pallier le manque de données nécessaires à la représentation numérique et s'avérer suffisante pour des applications de surveillance ou de diagnostic pour lesquelles la chronologie des événements est essentielle. (Shults *et al.*, 1997) proposent d'utiliser l'outil de simulation qualitative QSIM pour vérifier des propriétés exprimées

en logique temporelle CTL* (Clarke *et al.*, 1986). En biologie, des outils de model-checking (HyTech pour (Ahmad *et al.*, 1991) et NuSMV pour (Batt *et al.*, 2010)) sont appliqués sur des modèles qualitatifs de réseaux de régulation génique pour en analyser les propriétés. Dans cette approche nous proposons d'utiliser les techniques de model-checking, non plus pour analyser uniquement des propriétés du système mais surtout pour apporter des réponses quant au comportement du système face à l'application d'un scénario. Cette approche, décrite dans (Largouët *et al.*, 2010), a pour originalité d'explorer le système dynamique sans avoir recours à la simulation. Les problèmes de prédiction sont en effet réexprimés sous la forme de propriétés à vérifier sur le système. Ces propriétés sont vérifiées par une méthode de model-checking symbolique (Henzinger *et al.*, 1994) dont le principe consiste à représenter et à manipuler de manière concise de très grands ensembles d'états. Nous proposons un langage de requête de haut niveau s'appuyant sur la logique temporelle TCTL (Alur *et al.*, 1994) qui permet à un utilisateur d'exprimer, sous une forme similaire à une requête de base de données, le scénario qu'il veut explorer. La réponse à cette requête lui permet, non seulement, d'améliorer sa compréhension de l'écosystème mais, également, de connaître les effets à terme d'actions sur le terrain, de déterminer les valeurs critiques des variables impliquées lui permettant ainsi de mieux évaluer les actions possibles en termes de gestion et de gouvernance.

Dans cet article, nous nous focalisons sur la description du logiciel EcoMata qui implémente cette approche. Nous insistons sur un aspect peu traité dans la littérature et qui est pourtant difficile, à savoir comment faciliter la modélisation de l'écosystème, et plus particulièrement dans notre cas, la construction du réseau d'automates temporisés qui en décrit le comportement. Le logiciel EcoMata permet une construction simple, intuitive et interactive d'un réseau d'automates, son interrogation automatisée à l'aide des patrons de requêtes et une restitution des résultats de l'interrogation sous une forme conviviale. Nous illustrons notre approche dans le domaine halieutique.

Dans la section 2, nous décrivons de manière succincte notre approche qui s'appuie sur une modélisation qualitative de l'écosystème à l'aide d'automates temporisés, et sur un ensemble de patrons de requête qui permettent à un utilisateur d'analyser l'écosystème de manière interactive (pour plus de détails, voir (Largouët *et al.*, 2010)). Nous présentons ensuite, dans la section 3, l'application qui nous sert à illustrer notre propos. Dans la section 4, nous présentons le logiciel EcoMata en nous appuyant sur un exemple simplifié tiré du domaine de l'halieutique. Dans la section 5, nous nous focalisons sur la génération automatique du modèle qualitatif sous forme d'un réseau d'automates. L'algorithme permettant cette génération, qui s'appuie sur le modèle Lotka-Volterra, est décrit de manière assez détaillée. Dans la section 6, nous décrivons les résultats expérimentaux et faisons une analyse des temps de traitement obtenus en distinguant trois catégories d'automates selon leur complexité. Nous concluons en section 7.

2. Formalisme de représentation et d'interrogation

Cette section introduit, dans un premier temps, les automates temporisés, formalisme adopté pour la représentation d'un écosystème, puis le model-checking dans un deuxième temps. Les techniques de model-checking (Clarke *et al.*, 2002) et en particulier les techniques de model-checking symboliques (Henzinger *et al.*, 1994; Yovine, 1998) permettent la représentation concise d'un ensemble d'états, par opposition à la représentation exhaustive. Nous proposons d'explorer le comportement d'un écosystème par la vérification de propriétés TCTL par ces techniques de model-checking symboliques, ce qui évite de devoir explorer l'espace des états de manière systématique. À partir de la syntaxe TCTL, nous proposons un langage de haut niveau sur lequel peuvent s'appuyer les utilisateurs, ce qui leur évite l'expression des requêtes en logique. Les patrons de requêtes sont présentés dans la dernière partie de cette section.

2.1. *Automate temporisé*

Dans cet article nous proposons de modéliser un écosystème et ses perturbations par des automates temporisés (Alur *et al.*, 1994). Les automates temporisés reprennent le formalisme des automates auxquels s'ajoutent des horloges. Dans le formalisme des automates, un système est représenté par un ensemble de sommets décrivant les états du système, ces sommets étant reliés par des arcs étiquetés par des événements. Dans un automate temporisé, la dynamique temporelle est décrite grâce aux horloges qui permettent la définition de contraintes temporelles associées aux sommets ou aux arcs. Une contrainte temporelle associée à un sommet est appelée son *invariant*, celle associée à une transition est nommée une *garde*. Il est possible de rester dans un état aussi longtemps que son invariant est vrai. Une transition peut être déclenchée dès que sa garde est vraie. Lors de l'exécution d'une transition, la remise à zéro des horloges est possible.

On note $\Phi(\mathcal{X})$ l'ensemble des contraintes d'horloges. Un automate temporisé est défini par un n-uplet
$< \mathcal{S}, \mathcal{X}, \mathcal{L}, \mathcal{E}, \mathcal{I} >$ où :

– $\mathcal{S}$ est un ensemble fini de sommets, $s_o \in S$ étant le sommet initial ;

– $\mathcal{X}$ est un ensemble fini d'horloges ;

– $\mathcal{L}$ est un ensemble fini d'étiquettes ;

– $\mathcal{E}$ est un ensemble fini d'arcs ; chaque arc e est un n-uplet $(s, l, \varphi, \delta, s')$ tel que e relie $s \in \mathcal{S}$, état source, à $s' \in \mathcal{S}$, état destination, et est étiqueté par l'événement l ; la condition d'activation que doit satisfaire les horloges pour franchir la transition est représentée par la contrainte temporelle $\varphi \in \Phi(\mathcal{X})$, $\delta \subseteq \mathcal{X}$ correspond à l'ensemble des horloges réinitialisées lorsque la transition est tirée ;

– $\mathcal{I} : S \rightarrow \Phi(\mathcal{X})$ associe à chaque sommet de l'automate temporisé une contrainte temporelle appelée l'*invariant* du sommet.

La théorie des automates temporisés permet de décrire un système complexe comme un produit d'automates, ce qui autorise la construction modulaire du système global. Chaque automate décrit un sous-système et la synchronisation entre les automates se fait par le biais d'*événements de synchronisation*. Lors de la définition des composants, il est alors nécessaire de distinguer les événements internes décrivant la dynamique propre et asynchrone du sous-système, des événements externes qui provoquent l'évolution simultanée de tous les sous-systèmes partageant ces événements.

2.2. *Model-checking*

Issu du domaine des méthodes formelles, le model-checking (Clarke *et al.*, 2002) est utilisé pour la vérification automatique de systèmes complexes représentés par des systèmes à événements discrets. Lorsque le modèle d'un système est décrit sous la forme d'un automate temporisé, les propriétés sont exprimées à l'aide d'une logique temporelle. Le problème du model-checking peut alors se résumer de la façon suivante : étant donné M un modèle du système et φ une propriété à vérifier, est-ce que M satisfait φ ? Les programmes de model-checking retournent une réponse binaire (la propriété φ est vérifiée ou ne l'est pas).

La logique la plus répandue pour la vérification d'automates temporisés est la logique TCTL (*Timed Computation Tree Logic*) qui est une extension de la logique CTL autorisant l'expression de contraintes temporelles dans les propriétés. Les formules TCTL sont définies de manière inductive par la grammaire suivante définie dans (Henzinger *et al.*, 1994) :

$$f ::= p \mid x \in I \mid \neg f \mid f_1 \vee f_2 \mid \exists \Diamond_I f \mid \forall \Diamond_I f$$

avec $p \in P$ un prédicat de base, $x \in \mathcal{X}$ une horloge et I un intervalle de temps. On trouve les abréviations $\forall \Box f$ pour $\neg \exists \Diamond \neg f$ et $\exists \Box f$ pour $\neg \forall \Diamond \neg f$. Intuitivement l'opérateur $\exists \Diamond_I f$ signifie qu'il existe une exécution menant à un état satisfaisant la propriété f au temps $t \in I$. $\forall \Diamond_I f$ signifie que chaque exécution possède un état où la propriété f est valide au temps $t \in I$. $\forall \Box f$ signifie que tous les états sur tous les chemins d'exécution satisfont la propriété f. $\exists \Box f$ signifie qu'il existe une exécution telle que tous les états sur le chemin d'exécution satisfont la propriété f.

Un automate temporisé ayant la particularité de posséder un nombre infini de comportements liés à la valuation de ses horloges, on se trouve confronté au problème de l'explosion combinatoire. Afin de résoudre ce problème, le model-checking symbolique représente des ensembles d'états à l'aide de structures de données efficaces comme les diagrammes de décision binaire (encore appelés BDD pour *Binary Decision Diagram*), ou les matrices de bornes (DBM pour *Difference Bound Matrices*). Les algorithmes de model-checking symboliques travaillent sur ces ensembles d'états et déterminent, en fonction des propriétés à vérifier, soit l'espace atteignable, soit l'ensemble des états satisfaisant une formule TCTL (Berard *et al.*, 2001).

Le langage de description UPPAAL

Les outils les plus connus pour le model-checking symbolique sur les automates temporisés sont KRONOS (Yovine, 1997) et UPPAAL (Larsen *et al.*, 1997). Ce dernier régulièrement mis à jour par de nouvelles fonctionnalités et disposant d'une interface graphique agréable a été utilisé pour la réalisation de cette étude. UPPAAL étend le formalisme des automates temporisés présenté précédemment par de nouvelles caractéristiques et des notations particulières. La première extension concerne la définition possible de variables entières pouvant être associées à des états et modifiées lors du déclenchement de transitions. La seconde évolution touche à la synchronisation d'automates réalisée par le biais de *canaux de synchronisation*. Par exemple avec UPPAAL les notations $a!$ et $a?$ représentent respectivement des événements émis et reçus sur le canal a. Dans le système global, produit synchronisé de tous les automates, à chaque événement reçu doit correspondre un événement émis alors que les transitions non synchronisées correspondent à une évolution asynchrone du sous-système.

2.3. *Patrons de requêtes en TCTL*

Des patrons de requêtes permettant l'interrogation d'un écosystème ont été proposés dans (Largouët *et al.*, 2010). Ces requêtes constituent un langage de haut-niveau plus facile à appréhender par les utilisateurs. Les questionnements, correspondant à la simulation de scénarios prédictifs, sont de type : "étant donné une situation initiale et une politique appliquée, que va-t-il se passer si ? ". On appelle $situation\ S$ l'état global du système représenté par une valeur qualitative de la biomasse pour chacune des espèces (la situation pouvant définir l'état d'une seule espèce jusqu'aux états de toutes les espèces). Afin de bénéficier de toute l'efficacité des techniques de model-checking permettant l'exploration de systèmes temporels à taille réelle, ces patrons de requêtes ont été exprimés à l'aide de la logique TCTL et sont décrits dans le tableau 1. Pour chaque patron, on définit l'objectif de la requête, ses données en entrée, la formulation de la requête en logique TCTL et les résultats produits. Les patrons $Reachability$, $Safety$ et $Always$ prennent en entrée une situation et traduisent une seule propriété TCTL à vérifier par model-checking : la réponse retournée est donc classiquement $vrai$ ou $faux$. Les patrons $WhichStates$ et $WhichDate$ prennent en entrée respectivement une date t et une situation et implémentent chacun une itération de formules TCTL. Une information plus précise en réponse à la requête peut donc être retournée, c'est pourquoi les patrons $WhichStates$ et $WhichDate$ renvoient respectivement les situations possibles et une date.

3. Application à un écosystème marin

Nous présentons l'application du formalisme des automates temporisés à la modélisation d'un écosystème et l'usage des patrons de requêtes pour son exploration. Cette illustration est faite à travers un écosystème marin simplifié décrit dans la section 3.1.

Patron	Description	Formulation TCTL	Résultats
Atteignabilité (**Reachability**)	Teste si une situation S est possible à une date (éventuellement non précisée)	$\exists\Diamond(S \wedge chrono.t = t)$ $\exists\Diamond(S)$	vrai ou faux
QuelsEtats (**WhichStates**)	Recherche toutes les situations S_i à une date t	Propriété de type atteignabilité itération de la formule sur les S_i $\exists\Diamond(S_i \wedge chrono.t = t)$	Les situations S_i au temps t
QuelleDate (**WhichDate**)	Recherche la date t_i de la première occurrence d'une situation S (depuis S_{init})	Propriété de type atteignabilité itération de la formule sur les t_i $\exists\Diamond(S \wedge chrono.t = t_i)$	Si S est atteignable, retourne le premier t_i
Sûreté (**Never**)	Vérifie si une situation indésirable S ne peut pas se produire	$not(\exists\Diamond S)$ or $\forall\Box(not\ S)$	vrai si S n'arrive jamais, faux sinon
Toujours (**Always**)	Vérifie qu'une situation S est toujours vérifiée	$\forall\Box(S)$	vrai si S est toujours satisfaite, faux sinon

Tableau 1. *Les cinq patrons de scénario définis pour explorer l'écosystème*

3.1. *Réseau trophique de l'écosystème*

Nous illustrons notre approche dans le domaine halieutique. Le modèle d'un écosystème marin soumis à des pressions de pêche est décrit dans la figure 1. Un écosystème récifo-lagonaire de l'atoll d'Uvéa en Nouvelle-Calédonie a été modélisé selon cette approche (Largouët *et al.*, n.d.). Des connaissances issues de précédentes études, ont permis de représenter cet écosystème de manière qualitative (cinq groupes trophiques et trois forces de pêches) puis d'étudier l'impact d'un accroissement de la pêche (l'écosystème n'étant aujourd'hui touché que par une pêche de subsistance).

Le modèle, présenté dans cet article, a été volontairement simplifié dans un but pédagogique. Dans cet exemple, l'écosystème est représenté par un réseau trophique composé de quatre espèces de poisson (le thon, le maquereau, la sardine et l'anchois), deux pressions de pêche (sur le thon et le maquereau) et une perturbation environnementale (le réchauffement climatique). Les espèces sont considérées comme des compartiments trophiques qui échangent des flux de biomasse *via* un processus de prédation. Le flux de biomasse entre la proie maquereau et le prédateur thon est représenté par la flèche reliant les deux espèces.

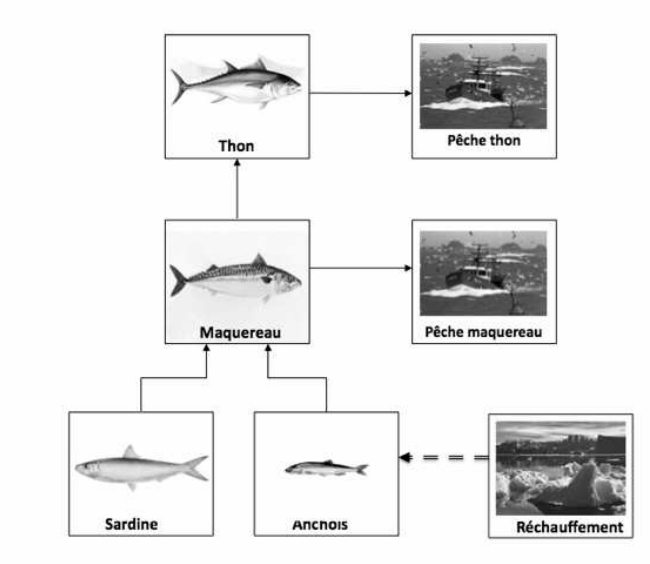

Figure 1. *Système halieutique simplifié : l'écosystème thon-maquereau*

3.2. *Modélisation de l'écosystème*

La modélisation proposée s'appuie sur une approche générique (Largouët *et al.*, 2010) dans laquelle l'écosystème est représenté de manière modulaire comme un ensemble d'espèces interagissant autour duquel se synchronisent d'autres modules représentant les perturbations (anthropiques ou naturelles). La modélisation est qualitative et permet de représenter à la fois le réseau trophique et les modules externes. L'intérêt de l'approche est de pouvoir représenter toutes les entités de l'écosystème dans un formalisme unifié permettant l'expression des contraintes temporelles propres à chaque sous-système. Un réseau trophique de type proie-prédateur se prête bien à la modélisation qualitative par systèmes à événements discrets grâce à : i) la discrétisation possible de ses valeurs de biomasse en niveaux de type *Low*, *Normal* ou *High* et ii) la possibilité de représenter la dynamique liée à l'interaction proie-prédateur par des événements agissant sur l'évolution des stocks.

Le comportement de l'écosystème est représenté par quatre automates, chacun étant associé à une espèce du réseau trophique. Pour chacune des espèces, l'automate décrit l'évolution de la biomasse entre les trois niveaux qualitatifs (*Low*, *Normal* et *High*). L'évolution entre ces niveaux est graduelle (impossibilité de passer du niveau *Low* au niveau *High* directement par exemple), c'est pourquoi on appelle les niveaux directement accessibles depuis un état, les états *voisins*. Le changement d'état d'une espèce est déclenché par un événement de synchronisation émis par un automate en interaction avec cette espèce (proie, prédateur ou pression de pêche). La transition

entre deux niveaux de biomasse n'est pas immédiate, elle est progressive avec un délai dépendant de l'espèce et du type de pression. On appelle *états stables*, les états de la biomasse qui n'évoluent pas avec le temps (en blanc sur la figure) et *états intermédiaires* (en noir) les états représentant une évolution entre deux valeurs du stock. Le délai de l'évolution est représenté, en tenant compte de l'incertitude liée à la connaissance des espèces, par la garde partant de l'état intermédiaire et l'invariant qui figure sur ce même état. Une fois le délai écoulé, le système atteint un nouvel état de biomasse. Il déclenche un événement de synchronisation indiquant son entrée dans ce nouvel état de biomasse. Cet événement pouvant déclencher à son tour des évolutions dans les autres automates.

La modélisation permet de représenter des annulations ou des changements de durée d'évolution (en cas de pression double, par exemple, comme l'augmentation d'un prédateur cumulée dans un deuxième temps à une forte pression de pêche). Des transitions entre états intermédiaires doivent donc être possibles, ainsi que des retours vers les états de biomasse d'origine. Les retours se font sans délai.

L'application du formalisme est illustrée pour l'espèce *thon* subissant la pression *pêche_thon* (cf. figure 2). L'état initial, représenté par un double cercle est l'état *Normal*. Lorsque la pression de pêche diminue (événement *peche_thon_low ?*), le système rentre dans un état intermédiaire dont la contrainte temporelle est *t<=40*. Lorsque le thon atteint la biomasse *High*, il génère un événement *thon_high !* qui peut modifier par synchronisation l'évolution de sa proie, le maquereau, dont le modèle n'est pas donné ici. L'annulation de cette diminution de la pression de pêche (représentée par l'événement *peche_thon_normal ?*) provoque le retour de l'espèce dans son état initial (*Normal*). Au contraire, si la pression de pêche est réajustée et passe de faible à nulle (événement *peche_thon_stop ?*), la transition vers l'état *High* est accélérée (transition entre états intermédiaires).

Les forces de pêches comme les pressions naturelles sont également représentées par des automates décrivant des niveaux qualitatifs et des contraintes de temps décrivant la durée de la pression ou le délai d'activation du réchauffement climatique.

3.3. *Exploration de l'écosystème*

Une fois le modèle de l'écosystème construit, on souhaite l'interroger selon différents scénarios de pêche en y ajoutant des perturbations naturelles éventuelles. Sur notre exemple, on peut imaginer un premier scénario qui simule l'augmentation de la pression de pêche sur nos deux espèces *thon* et *maquereau*. Cette forte pression de pêche peut être de durée limitée, 24 mois pour la pêche du thon et le double pour la pêche du maquereau. Les patrons de requêtes sont alors invoqués afin d'observer le comportement de l'écosystème dans le temps. Par le patron *WhichStates*, on cherche à savoir quels sont les états possibles du système pour diverses dates (de 1 à 4 ans par exemple). Puis, pour une ou plusieurs situations particulières intéressant l'utilisateur, il est alors possible de connaître la date de leur première occurrence grâce au patron

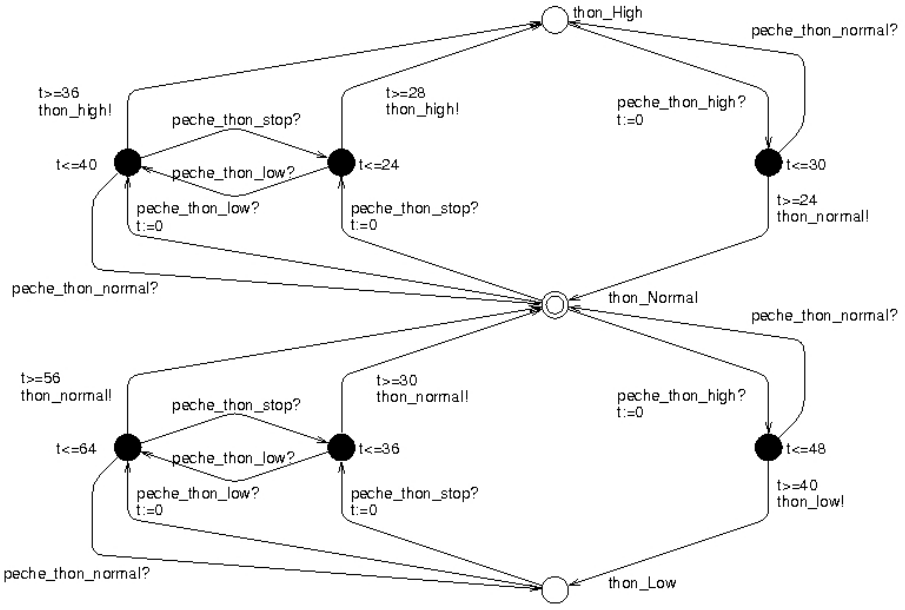

Figure 2. *Exemple d'automate modélisant l'espèce thon*

WhichDate. Le scénario *Never* permet de savoir si une situation indésirable peut se produire ou non. Dans notre exemple, on peut imaginer que l'écosystème qui présenterait les biomasses de toutes ses espèces à un niveau *Low* serait critique. Le patron *Never* permet de décrire cette situation et de vérifier si elle peut se produire un jour. Dans l'affirmative, une requête de type *WhichDate* donnera la date de la première occurrence. Le patron *Always* sera utilisé lorsque l'utilisateur souhaite s'assurer qu'une propriété de l'écosystème est toujours satisfaite (un niveau de biomasse minimal pour une espèce donnée par exemple).

4. Logiciel EcoMata

Étant donné la complexité du réseau d'automates qui augmente exponentiellement en fonction de nombre d'espèces et du nombre de pressions, la construction manuelle du modèle devient une tâche impossible. EcoMata est un logiciel permettant la construction automatique des automates et la définition interactive des pressions de pêche que l'on souhaite appliquer sur l'écosystème. Une fois le modèle construit, les requêtes peuvent être soumises à l'aide d'une interface graphique. Cette section présente une vue globale du logiciel EcoMata et son utilisation sur l'exemple simplifié de l'écosystème.

Le logiciel est composé de trois parties principales (cf. figure 3) :

– l'*éditeur d'écosystème* propose une interface graphique permettant aux utilisateurs de définir tous les éléments de l'écosystème : les espèces avec leurs paramètres biologiques, les pressions ainsi que leurs interactions ;

– le *générateur d'automate* construit, à partir de la définition de l'écosystème, d'une topologie par défaut des espèces et des pressions de pêche, le réseau d'automates synchronisés au format d'UPPAAL ;

– le *lanceur de requêtes* propose une interface graphique pour que l'utilisateur choisisse le patron de requête et définisse ses paramètres en entrée. Le lanceur interprète ce scénario en langage TCTL et exécute le model-checker d'UPPAAL en fournissant la configuration d'automate et la formule de requête. À la fin de la vérification, le lanceur récupère les résultats et les traces d'exécution puis les affiche sous une forme textuelle ou graphique.

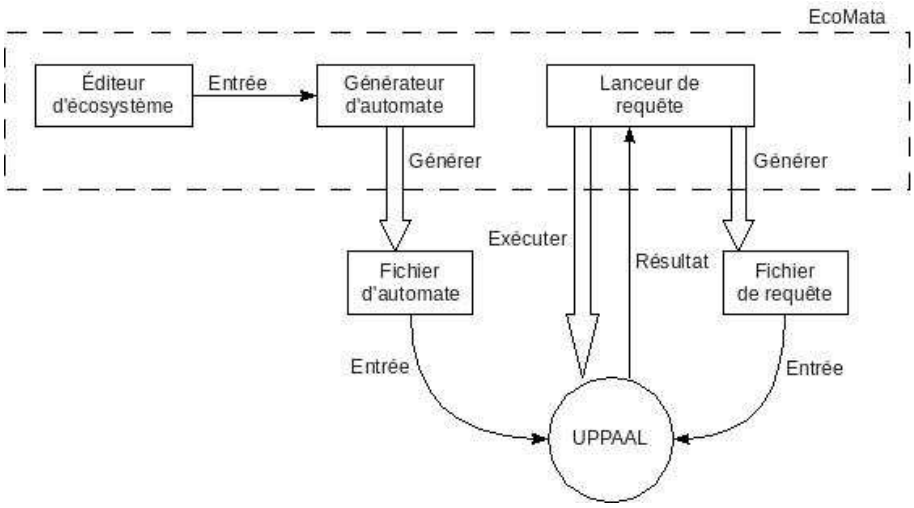

Figure 3. *Architecture du logiciel : les trois composants et ses relations avec le model-checker* UPPAAL

4.1. *Éditeur d'écosystème*

Pour illustrer la construction de l'écosystème à l'aide d'EcoMata, nous reprenons l'exemple du réseau trophique marin thon-maquereau présenté section 3.2. Dans notre exemple, l'unité de temps est l'année mais le pas de temps peut être défini, au choix, par l'utilisateur. Pour chaque espèce de poisson, trois paramètres biologiques sont requis :

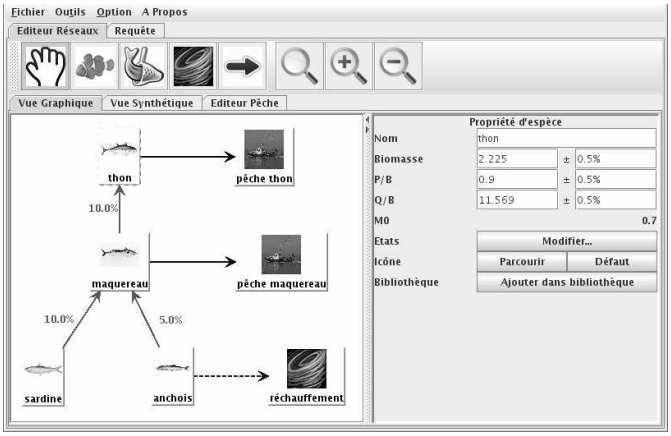

Figure 4. *Création du réseau trophique de l'écosystème thon-maquereau à l'aide de l'éditeur d'écosystème d'EcoMata*

– B est la biomasse d'une espèce à l'équilibre. Cette valeur correspond à l'état qualitatif *Normal*. L'unité par défaut est $tonne/km^2$;

– P/B est le taux de croissance de la biomasse, en absence de prédateur et avec suffisamment de proies. Le taux est annuel ;

– Q/B est le taux de consommation de la biomasse. C'est la quantité d'aliment nécessaire pour que la biomasse d'une espèce puisse croître à la vitesse de P/B. Ce taux est également annuel.

Les paramètres biologiques de l'écosystème thon-maquereau sont donnés dans le tableau 2. Pour un écologue, spécialiste d'un site d'étude, ces informations sont généralement bien connues. Le nombre d'états qualitatifs par défaut est de quatre (*Danger*, *Low*, *Normal* et *High*) mais l'utilisateur a la possibilité d'en définir le nombre qu'il souhaite. Chaque niveau correspond à une proportion du niveau *Normal* et est variable selon les espèces. Pour créer une nouvelle espèce, l'utilisateur peut, sur l'interface graphique, faire glisser une des espèces existant déjà dans la bibliothèque d'espèces ou la créer de toute pièce en saisissant les paramètres biologiques présentés ci-avant. Chaque nouvelle espèce créée peut enrichir la bibliothèque d'espèces ou être exportée afin d'alimenter éventuellement d'autres écosystèmes. Afin de rendre plus explicite le réseau trophique, les espèces sont représentées par des icônes dont les images sont laissées au libre choix de l'utilisateur. Dans l'éditeur d'écosystème, les pressions de pêche sont caractérisées par le pourcentage de biomasse capturée de l'espèce pêchée. Elles sont également définies par des niveaux qualitatifs, dont le nombre, la qualification et le rapport (au niveau normal) sont donnés par l'utilisateur.

Les interactions entre les espèces sont définies par le taux de prédation (DC) entre les prédateurs et les proies. Dans le logiciel, cette valeur est définie en cliquant sur le

Groupe	$B\ (tonne/km^2)$	$P/B(par\ an)$	$Q/B(par\ an)$
Thon	2,225	0,9	11,569
Maquereau	6,27	0,823	14,2
Sardine	13,79	2,74	14,0
Anchois	23,95	2,898	13,5

Tableau 2. *Paramètres biologiques de chacune des espèces de l'écosystème*

lien symbolisant le transfert de biomasse. Pour notre écosystème, les proportions de prédations sont les suivantes :

– $10\ \%$ de l'alimentation du thon est constituée de maquereau,

– $10\ \%$ de l'alimentation du maquereau est constituée de sardine et $5\ \%$ d'anchois.

Les autres sources d'alimentation des espèces représentées sont prises en considération dans le modèle écologique sous-jacent et ne doivent pas nécessairement être représentées dans le réseau trophique. En effet, il est souvent plus simple (et plus efficace du point de vue informatique) de limiter la définition de l'écosystème aux espèces touchées par la pêche ou la conservation.

Pour compléter la définition succincte de la pression de pêche donnée dans le réseau trophique, un onglet particulier appelé *éditeur de pêche* permet à l'utilisateur de définir précisément ses stratégies de pêches dans le temps. Deux modes de définition des pressions de pêche sont possibles :

– schématisé : à l'aide d'un chronogramme, l'utilisateur définit la durée de la pression pour chaque niveau qualitatif de pression de pêche ;

– auto-adaptatif : la pression de pêche est guidée par le niveau de la biomasse. Ainsi pour chaque niveau atteint, on définit la pression de pêche correspondante (par exemple lorsque le niveau de la biomasse atteint un niveau *Low*, on réduit la pression de pêche).

L'intérêt de l'onglet *éditeur de pêche* est qu'il permet à l'utilisateur de réviser facilement ses scénarios en fonction des résultats obtenus lors d'une précédente exécution. EcoMata peut alors être utilisé de manière interactive comme un outil d'aide à la décision.

Dans notre réseau trophique thon-maquereau, nous définissons les pressions de pêche à l'aide du mode schématisé (ou chronogramme). À l'état *Normal*, nous prélevons 20 % de la biomasse pour le thon et pour le maquereau. Ce taux est doublé à l'état *High*. Le thon est pêché avec le niveau *High* pendant 12 années avant que la pêche ne soit arrêtée (état *Stopped*). Le maquereau qui est pêché à un niveau *Normal* pendant les 12 premières années de la simulation et est pêché au niveau *High* à partir de la treizième année. Ce cycle de 24 années est ensuite reproduit à nouveau. La perturbation *Réchauffement* simule le phénomène d'*el niño* qui arrive une fois tous les 4

ans. Ce réchauffement cause une diminution de 5 % de la biomasse d'anchois. La définition de l'évolution des niveaux de pressions (pêche, environnement) est présentée dans la figure 5.

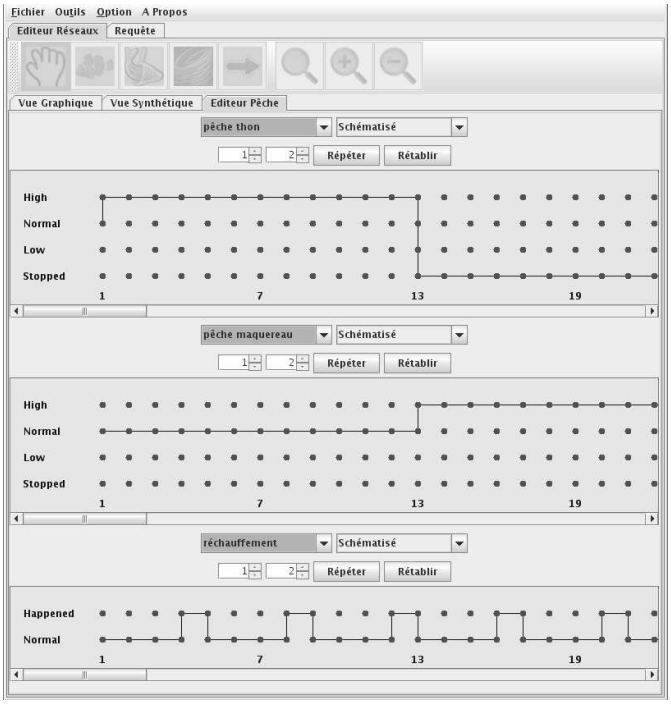

Figure 5. *Définition des pressions de pêche à l'aide d'un chronogramme. Alternance des pressions aux niveaux High et Stopped pour le thon et Normal et High pour le maquereau*

4.2. *Générateur d'automates*

Une fois le réseau trophique défini, nous pouvons lancer le générateur pour construire un ensemble d'automates temporisés à partir des données du réseau trophique et de la topologie des espèces selon les niveaux qualitatifs définis (Largouët *et al.*, 2010). Dans EcoMata, la représentation sous forme d'automates n'est pas directement accessible à l'utilisateur qui ne manipule que le réseau trophique, les chronogrammes de pressions de pêche ou les patrons de requêtes. Il est cependant nécessaire de générer le réseau d'automates pour résoudre les requêtes traduites en TCTL par model-checking. L'algorithme de génération de l'automate est décrit section 5.

4.3. *Lanceur de requête*

L'algorithme de génération informe l'utilisateur de la complexité de l'automate généré et des délais attendus pour l'exécution de ses requêtes. Un menu déroulant permet d'accéder aux cinq patrons de requêtes ainsi qu'à une requête de synthèse (résumant les états de toutes les espèces sur la durée de la simulation). Cette section présente quelques exemples d'interrogation sur notre réseau trophique thon-maquereau.

La première requête que nous souhaitons poser est : « Les stratégies de pêche définies sur le thon et le maquereau vont-elles conduire la biomasse du maquereau à l'état *Danger* ? » La requête *Never* répond aux questions de type sûreté. Le résultat de cette requête, produit en quelques secondes, nous informe que cette situation « peut arriver » (cf. figure 6).

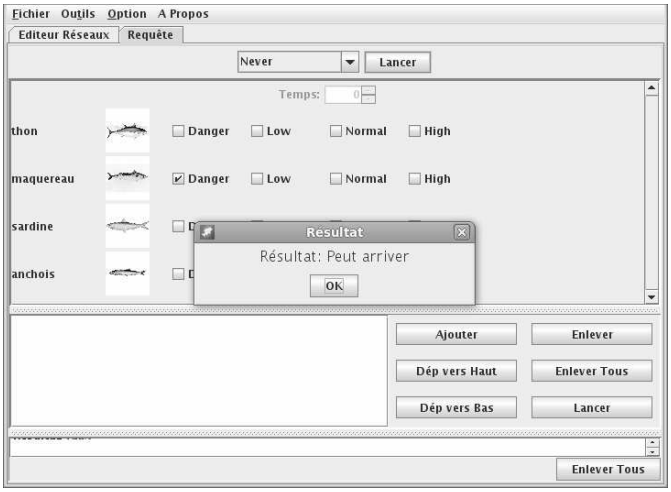

Figure 6. *Requête de type « Never » invoquée pour une situation où le maquereau est à l'état Danger*

Étant donné la réponse précédente, nous voulons savoir à quelle date cette situation peut se passer au plus tôt. La requête *WhichDate*, pour laquelle, on coche simplement comme paramètre d'entrée, le maquereau à l'état *Danger*, nous informe que cette situation peut se produire en 26 ans (cf. figure 7).

Pour connaître l'état du système à cette période, nous lançons la requête *WhichStates* à la date 30. La figure 8 présente le résultat de cette requête sous la forme d'une liste de situations possibles. Un bouton au bout de chaque situation permet d'ouvrir une nouvelle fenêtre dans laquelle figure une trace possible menant à cette situation (voir la trace menant à la *situation 1* dans la figure 9).

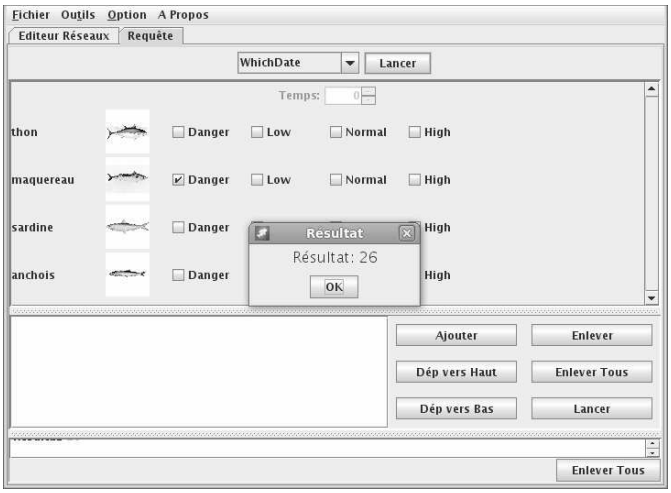

Figure 7. *Requête du type « WhichDate » invoquée pour une situation où le maquereau est à l'état Danger*

Figure 8. *Résultat de la requête « WhichStates » invoquée pour la date 30 années*

5. Génération automatique d'un réseau d'automates

Afin de rendre plus facile l'élaboration du réseau d'automates temporisés et dans le cas d'un écosystème pour lequel on dispose d'un modèle numérique, ne serait-ce que partiel, nous avons développé un algorithme générique permettant de générer ce réseau à partir de la définition d'un réseau trophique. Comme décrit dans la section précédente, les états qualitatifs d'espèce sont représentés par les états d'un automate. La garde sur les transitions et les invariants sur les états permettent d'exprimer les contraintes temporelles associées à l'évolution de la biomasse de l'espèce. Dans le cas de notre application, nous avons choisi de nous appuyer sur le modèle numérique Lotka-Voltera.

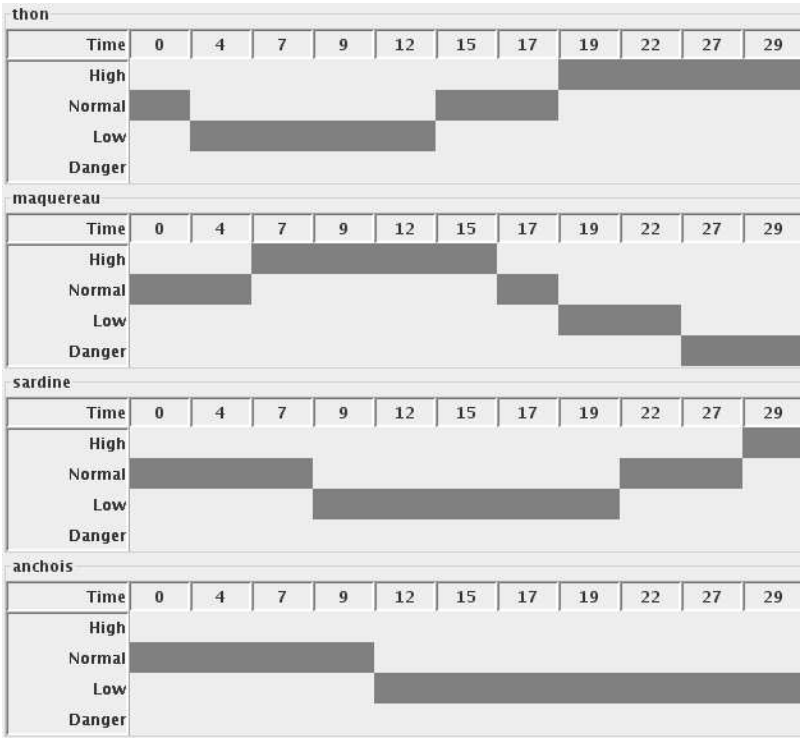

Figure 9. *Trace menant à la situation 1 (thon : High, maquereau : Danger, sardine : High, anchois : Low)*

5.1. *Modèle numérique Lotka-Voltera*

Les équations de Lotka-Volterra, que l'on désigne aussi sous le terme de « modèle proie-prédateur » , sont un couple d'équations différentielles non linéaires du premier ordre. Elles sont couramment utilisées pour décrire la dynamique de systèmes biologiques dans lesquels un prédateur et sa proie interagissent (Murray, 2002).

$$\begin{cases} N(0) = N_0,\ P(0) = P_0, \\ \dfrac{\mathrm{d}N(t)}{\mathrm{d}t} = N(t)(\alpha - \beta P(t)) \\ \dfrac{\mathrm{d}P(t)}{\mathrm{d}t} = P(t)(\delta P(t) - \gamma) \end{cases}$$

où :

– $N(t)$ est l'effectif des proies, $P(t)$ l'effectif des prédateurs, N_0 la population initiale des proies, et P_0 la population initiale des prédateurs ;

– t est le temps, $\frac{\mathrm{d}N(t)}{\mathrm{d}t}$ et $\frac{\mathrm{d}P(t)}{\mathrm{d}t}$ représentent la croissance des populations au cours de temps,

– α est le taux de reproduction des proies en absence de prédateur, β le taux de mortalité des proies due aux prédateurs, δ est le taux de reproduction des prédateurs en fonction des proies mangées, et γ le taux de mortalité des prédateurs en l'absence de proies.

Comme expliqué en section 3.2, nous modélisons l'écosystème par un système à événements discrets et la valeur de la biomasse de chaque espèce est discrétisée par des états qualitatifs. Les évolutions de la biomasse, qui ont une certaine durée, sont représentées par le passage d'un état qualitatif dit stable à un état qualitatif voisin stable, au travers d'un état dit intermédiaire ; les changements d'un état à un autre sont eux instantanés.

Les équations de Votka-Volterra sont utilisées pour évaluer le temps de séjour dans l'état intermédiaire, et donc les contraintes temporelles étiquetant les transitions qui permettent d'y arriver et d'en sortir, et celles (invariant) étiquetant l'état intermédiaire lui-même. Un état intermédiaire décrit l'évolution de la biomasse d'une espèce, en supposant l'absence de changement de biomasse des espèces voisines[1]. Le temps de séjour peut donc être calculé en considérant les biomasses des espèces voisines inchangées. Ainsi, les deux équations différentielles sont réduites à une seule. En effet, la valeur de $P(t)$ reste constante lors du calcul de $N(t)$. L'avantage le plus important de cette hypothèse est que l'équation peut alors être résolue de manière formelle, ce qui évite une résolution numérique coûteuse en temps, et raccourcit de manière remarquable le temps de génération des automates.

5.2. *Algorithme*

Nous illustrons le processus de génération d'automates. Il commence par la construction proprement dite des automates, successivement pour chacune des espèces. Cela inclut la création des états qualitatifs, stables et intermédiaires, et des transitions entre ces états. Cette étape utilise les équations de Lotka-Volterra pour identifier les changements et les contraintes temporelles associées.

La génération automatique conduit en général à un ensemble d'automates dont le nombre d'états et de transitions est important. Afin de diminuer la complexité du modèle, et d'améliorer en conséquence le temps de réponse aux requêtes, il est possible ensuite de simplifier le modèle en regroupant les états similaires, afin de réduire le nombre d'états et de transitions des automates obtenus. Ce regroupement utilise un algorithme de classification hiérarchique ascendante. Le paramétrage de l'algorithme permet de contrôler de plus ou moins grands nombres de regroupements.

1. La prise en compte de changements de biomasse des espèces voisines pendant la durée de l'évolution représentée par un état intermédiaire est traitée par l'existence de transitions transversales comme on le voit en 5.2.

L'algorithme est illustré sur l'exemple d'un écosystème n'ayant que trois espèces : $sp0 \rightarrow sp1 \rightarrow sp2$. L'espèce *sp0* est la proie de l'espèce *sp1* et l'espèce *sp2* est le prédateur de l'espèce *sp1* (les flèches représentent le transfert de biomasse). Les deux espèces *sp0* et *sp2* sont dites voisines de l'espèce *sp1*. Nous nous focalisons dans la suite sur la génération de l'automate de l'espèce *sp1*.

5.2.1. *Construction de l'automate espèce par espèce.*

La construction de l'automate d'une espèce passe par la construction des états qualitatifs stables, puis la construction des états qualitatifs intermédiaires, et finalement la construction des transitions étiquetées par des conditions et des contraintes temporelles.

L'algorithme de construction automatique des automates est schématisé dans la figure 10.

– *Construction des états stables :* chaque espèce est à l'équilibre dans un état qualitatif stable. Ces états sont décrits par des valeurs qualitatives, dans notre cas celles de la biomasse de l'espèce. Pour l'espèce *sp1*, les états stables sont : *sp1_H, sp1_N, sp1_L*, correspondant aux trois valeurs qualitatives de biomasse, *H* (*High* pour élevé), *N* (*Normal*), et *L* (*Low* pour bas).

– *Construction des états intermédiaires :* lorsqu'un changement d'état se produit dans une espèce voisine de *sp1* (dans le réseau trophique), ici *sp0* ou *sp2*, l'espèce *sp1* peut changer d'état qualitatif, en passant par un état dit intermédiaire, représentant la période d'évolution entre deux états stables. Il faut donc identifier toutes les combinaisons d'états des espèces voisines, qui peuvent faire évoluer l'espèce et la mener dans un tel état intermédiaire. À partir de l'état *sp1_N* par exemple, le tableau 3 présente 4 possibilités parmi 9, en indiquant la tendance (stable, croissante, décroissante) de l'évolution de la biomasse d'un état vers un autre, et sa durée. Un état intermédiaire est créé dans tous les cas où la tendance n'est pas « stable ».

La première ligne du tableau 3 correspond au cas où soit *sp0*, soit *sp2* arrive dans l'état *L (Low)* alors que l'autre y est déjà[2]. La tendance pour *sp1* est alors de rester stable et aucun état intermédiaire n'est créé. La troisième ligne correspond au cas où *sp0* arrive dans l'état *H (High)* alors que *sp2* était dans l'état *L (Low)*, ou celui où *sp2* arrive dans l'état *L (Low)* alors que *sp0* était dans l'état *H (High)*. La tendance est alors croissante pour *sp1*, avec une durée d'évolution entre *N (Normal)* et *H (High)* comprise entre 23 et 36 unités de temps. Un état intermédiaire est donc créé, sous le nom *sp1_N_sp0_H_sp2_L_UP*, obtenu par concaténation des noms des états des espèces et de la tendance.

– *Construction des transitions vers les états intermédiaires :* après avoir construit les états intermédiaires, examinons maintenant comment construire les transitions menant à ces états.

2. On suppose toujours que deux transitions ne peuvent pas être simultanées.

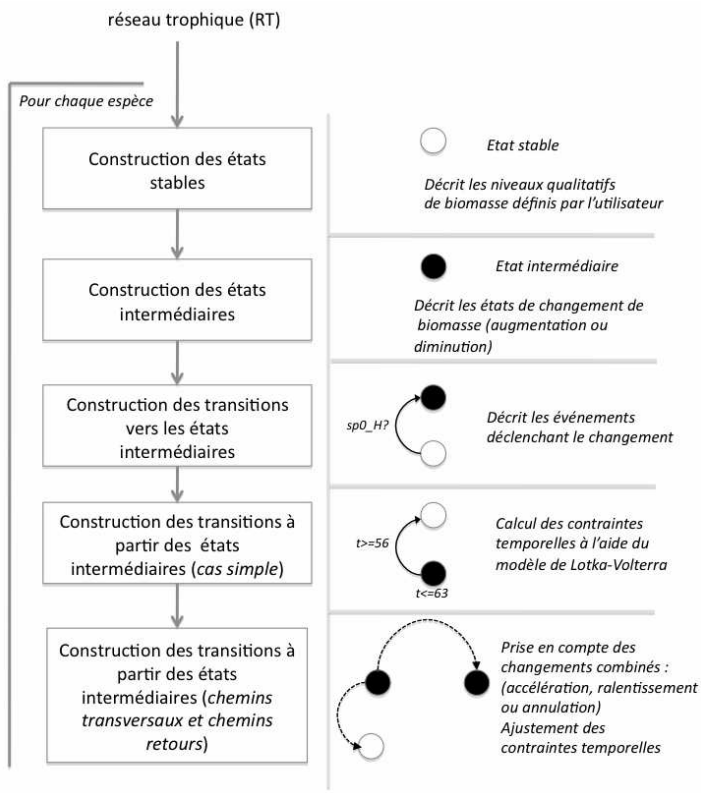

Figure 10. *La génération automatique des automates se fait espèce par espèce et comprend quatre étapes principales. La figure illustre le lien entre les étapes et leurs effets (ajout d'états, de transitions, de contraintes) sur les automates en cours de construction*

État de départ	États des esp. voisines	Tendance	Durée min	Durée max
sp1_N	*sp0_L sp2_L*	stable	-	-
sp1_N	*sp0_L sp2_H*	décroissante	56	62
sp1_N	*sp0_H sp2_L*	croissante	23	36
sp1_N	*sp0_H sp2_N*	croissante	56	63

Tableau 3. *États intermédiaires envisagés et leurs caractéristiques*

Ces transitions correspondent au changement de biomasse d'une espèce voisine, qui provoque l'évolution de la biomasse de l'espèce considérée, et donc un changement d'état vers l'état intermédiaire. L'état intermédiaire *sp1_N_sp0_H_sp2_L_UP*

peut être la cible de deux changements d'états, l'un provoqué par le changement de biomasse de *sp0* et l'autre par le changement de biomasse de *sp2*. Il s'agit des deux cas suivants : la proie *sp0* entre dans l'état « *High* » alors que le prédateur *sp2* est dans l'état « *Low* », ou le prédateur *sp2* entre dans l'état « *Low* » alors que la proie *sp0* est dans l'état « *High* ». On construit donc deux transitions, arrivant toutes deux dans l'état *sp1_N_sp0_H_sp2_L_UP*, étiquetées par l'événement de synchronisation déclenchant la transition, auquel on ajoute un *?*, et par la propriété qui doit rester vérifiée, appelée *garde*, soit :

Transition 1 : *sp0_H ?*, garde : *sp2==L* et Transition 2 : *sp2_L ?*, garde : *sp0==H*

– *Construction des transitions à partir des états intermédiaires (cas simple) :* il faut ensuite construire les transitions sortant des états intermédiaires vers l'état arrivée. Le cas simple est celui où l'évolution de la biomasse se produit sans qu'aucun autre changement ne perturbe le processus. L'algorithme fait de nouveau appel aux équations de Lotka-Volterra pour calculer les contraintes temporelles associées, ce qui permet d'obtenir l'invariant de l'état intermédiaire et les contraintes temporelles de la transition vers l'état d'arrivée. Cette transition est souvent associée à l'envoi d'un message de synchronisation, permettant de signaler le changement d'état de cette espèce aux autres espèces du réseau trophique ; la transition est alors étiquetée par un événement de type synchronisation terminée par un *!*.

Si on se reporte à la troisième ligne du tableau 3, une transition est créée qui sort de l'état intermédiaire *sp1_N_sp0_H_sp2_L_UP* et arrive à l'état *sp1_H*. On calcule l'invariant *t<=36* et on en étiquette l'état intermédiaire, puis on ajoute la synchronisation *sp1_H !* et la contrainte *t>=23* à la transition sortante.

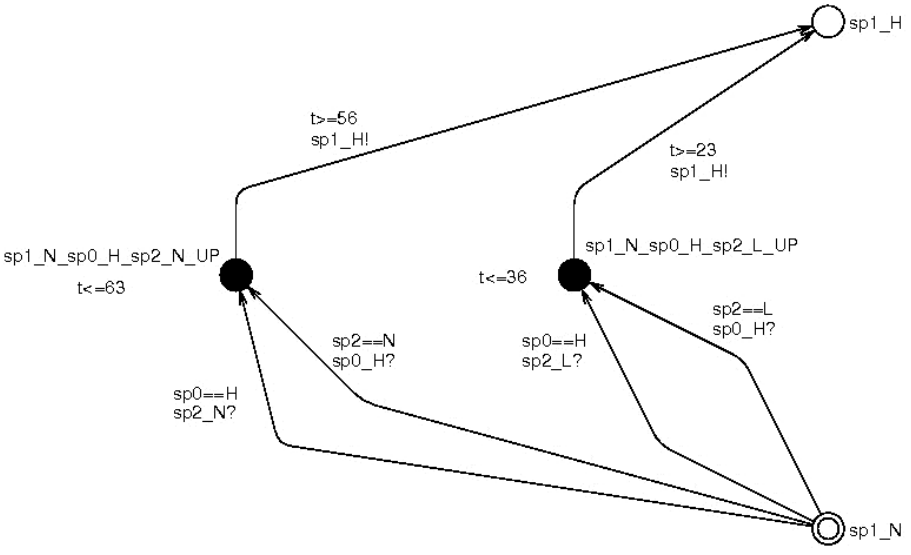

Figure 11. *Ajout des transitions vers les états intermédiaires*

La figure 11 illustre une partie de l'automate généré à la fin de ces quatre premières étapes. Les états blancs représentent les états qualitatifs stables de la biomasse de sp1. Les états noirs, dits états intermédiaires, représentent l'évolution de la biomasse de sp1. Les contraintes temporelles (t<=36 et t<=63) sur les états noirs représentent la durée maximale de l'évolution. Les transitions partant des états noirs et se dirigeant vers les états blancs correspondent à la fin de l'évolution. Ces transitions sont étiquetées par une contrainte temporelle représentant la durée minimale de l'évolution et par un éventuel événement de synchronisation (terminé par un !). Les transitions partant des états blancs et se dirigeant vers les états noirs correspondent au début d'une évolution. Les contraintes sur ces transitions sont les conditions de déclenchements.

– *Construction des transitions à partir des états intermédiaires (cas des chemins transversaux et des chemins de retour) :* lorsqu'une espèce est dans un état intermédiaire, elle y reste tant que les contraintes temporelles identifiées à l'étape précédente n'exigent pas d'en sortir. Ces contraintes temporelles expriment le temps nécessaire au passage entre deux états stables voisins, en absence de tout autre événement. Or, des changements sur les autres espèces peuvent accélérer, ralentir ou même interrompre l'évolution en cours, s'ils interviennent pendant cet intervalle de temps. Pour représenter ces interactions, il est nécessaire d'ajouter un certain nombre de transitions, permettant, soit le retour vers l'état stable avant le début de l'évolution en cas d'interruption, soit la transition vers des états dits tranversaux, intermédiaires eux aussi, qui permettent d'exprimer l'accélération ou le ralentissement de l'évolution par de nouvelles contraintes temporelles.

Puisque l'on ne peut passer en un seul changement que vers un état stable voisin, il suffit, à partir de l'état *sp1_N_sp0_H_sp2_L*, de rechercher l'existence des états intermédiaires : *sp1_N_sp0_H_sp2_N* et *sp1_N_sp0_N_sp2_L*. Si ces état ont déjà été construits lors de l'étape précédente, on ajoute une transition (chemin transversal) entre les deux états intermédiaires. C'est le cas du premier état *sp1_N_sp0_H_sp2_N*, et la nouvelle transition (en pointillé sur la figure 12) correspond à une évolution toujours croissante mais ralentie. Si ces états n'existent pas, on ajoute une transition retour qui ramène à l'état de départ. C'est le cas du second état *sp1_N_sp0_N_sp2_L*, et la nouvelle transition (en pointillé sur la figure 12) correspond à une interruption de l'évolution et ramène dans un état stable. Cette étape un peu technique est illustrée dans la figure 12. Les conditions étiquetant les transitions en pointillé ne sont pas données pour ne pas alourdir la figure.

5.2.2. *Simplification des automates*

Après avoir construit l'automate de chaque espèce, on procède à une simplification de ceux-ci.

Regroupement d'états intermédiaires : les états intermédiaires créés peuvent être nombreux et aussi être assez similaires. Or, le nombre d'états augmente nettement la complexité de résolution des requêtes par model-checking. Il est donc intéressant de regrouper les états, en particulier ceux qui sont similaires. On utilise pour cela

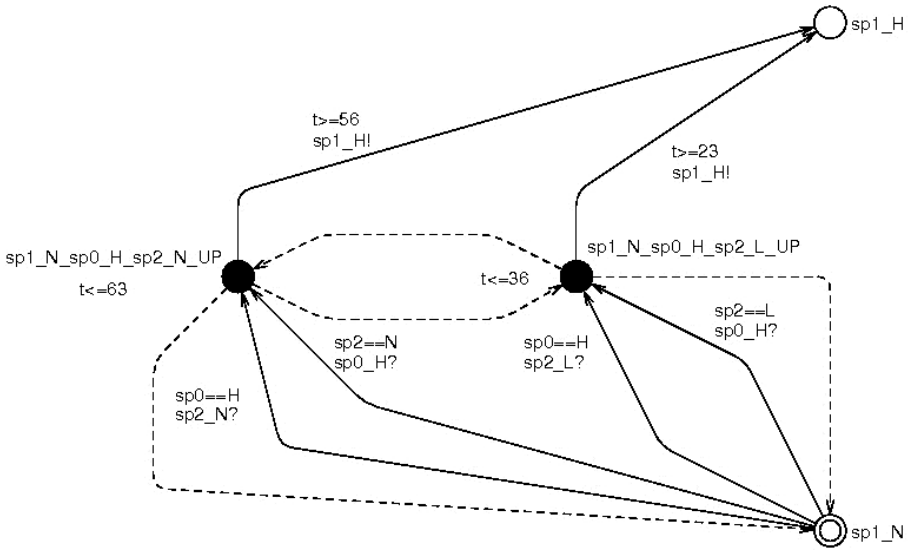

Figure 12. *Construction des chemins transversaux et des chemins de retour. Les chemins en pointillé correspondent aux événements qui modifient la vitesse d'évolution (accélération ou ralentissement selon le sens) ou aux événements qui interrompent l'évolution en cours*

un algorithme classique de classification hiérarchique ascendante. Cet algorithme regroupe automatiquement les états en groupes d'états similaires (ou *clusters*) en s'appuyant sur une « distance ». La distance d est calculée par la formule suivante : $d = \sqrt{\Delta I^2 + \Delta G^2}$ où ΔI (resp. ΔG) est la différence des invariants (resp. gardes) entre les états.

Il est possible de contrôler le regroupement en indiquant le nombre maximal d'états intermédiaires que l'on autorise entre deux états qualitatifs stables voisins (ou dit autrement, le nombre maximal de transitions que l'on autorise entre deux états qualitatifs stables voisins). Cette valeur est paramétrable par l'utilisateur. Plus on diminue ce paramètre, plus on perd en précision de la modélisation, mais plus on diminue la complexité de traitement d'une requête[3].

Regroupement de transitions : le regroupement d'états intermédiaires de l'étape précédente permet de supprimer certaines des transitions regroupées. C'est le cas lorsque les états regroupés vérifient les propriétés suivantes : les différences entre les états ne concernent qu'une seule espèce, et tous les états de l'espèce sont énumérés dans ce groupe.

3. Après une série d'expérimentations, la valeur 12 s'est avérée être un bon compromis, pour notre application, entre la précision du modèle et le temps de réponse des requêtes.

C'est le cas par exemple lorsque les trois états suivants sont regroupés : *sp1_N_sp0_L_sp2_H*, *sp1_N_sp0_N_sp2_H*, *sp1_N_sp0_H_sp2_H*.

En effet, les différences entre ces états ne concernent que l'espèce *sp0* et *sp0_L*, *sp0_N* et *sp0_H* sont toutes les valeurs qualitatives possibles de *sp0*. L'état de *sp0* peut donc être ignoré et la condition d'entrée dans cet état intermédiaire ne dépend plus de *sp0* ; il est ainsi possible regrouper trois transitions en une seule.

À l'issue de l'optimisation de certains automates initialement complexes, le nombre de transitions menant à un état intermédiaire est réduit de plus de 900 à moins de 300. Ceci réduit remarquablement la complexité du traitement des requêtes et en conséquence le temps d'interrogation.

6. Benchmarks (limites d'utilisation)

Afin d'évaluer les temps de réponses des requêtes soumises au model-checker d'UPPAAL, appelé VERIFYTA (Larsen *et al.*, 1997), qui croissent avec la complexité des automates, nous nous sommes intéressés à la requête de type *Reachability*. Cette requête est la composante de base des patrons plus compliqués que sont *WhichStates* et *WhichDate*. Les requêtes *Never* et *Always* sont plus simples et ne prennent que quelques secondes quelle que soit la taille du système.

Nb d'espèces	Nb de forces de pêche	Nb d'états	Nb de transitions	Nb d'états de l'automate le plus grand	Temps de réponse *Reachability*
1	1	55	92	51	0,245
2	1	357	654	357	0,664
3	1	783	2146	717	2,090
4	2	836	2237	717	2,038
5	2	1184	2867	717	2,472
6	2	1644	4410	777	3,523
7	2	1902	4894	777	4,3
8	2	3484	9316	777	6,093

Tableau 4. *Résultats expérimentaux sur des réseaux trophiques caractérisés par le nombre d'espèces et le nombre de forces de pêche : nombre d'états et de transitions du modèle global, nombre d'états de l'automate le plus grand du modèle et temps de réponse pour une requête de type Reachability*

Parmi les diverses caractéristiques des automates, nous n'avons conservé que celles qualifiant la complexité du modèle global. Ces caractéristiques sont, entre autres, les nombres d'états et de transitions du modèle (tous automates confondus) ainsi que le nombre d'états de l'automate le plus grand du réseau (en nombre d'états). Cette dernière variable n'augmente pas linéairement en fonction du nombre d'espèces

du réseau trophique et dépend fortement du nombre d'interactions entre les espèces. Le tableau 4 présente des réseaux trophiques pour lesquels le nombre d'espèces et de pressions de pêche augmentent au fil des exemples. Chaque espèce et chaque pression de pêche sont définies par quatre niveaux qualitatifs. À partir du troisième exemple, chaque espèce définit au moins trois interactions avec les autres éléments du réseau (espèces ou pressions de pêche).

S'appuyant sur ces résultats, EcoMata propose une classification du modèle généré selon trois catégories : peu complexe, moyennement complexe et très complexe. À l'issue de la génération des automates, un message informe l'utilisateur de la catégorie de son modèle et, en conséquence, des temps de réponse attendus.

7. Conclusion

Cet article présente le logiciel EcoMata, un outil d'aide à la décision pour la gestion et la sauvegarde des écosystèmes. L'approche originale implémentée par cet outil consiste à représenter l'écosystème par un système à événements discrets et à exploiter l'efficacité des techniques de model-checking pour répondre aux requêtes d'un utilisateur, non par simulation, mais par la vérification de propriétés sur l'écosystème. Le premier intérêt de cette approche est de proposer un formalisme unifié et qualitatif pour représenter tous les éléments constituant un écosystème. Cette approche générique permet de représenter un réseau trophique de type proie-prédateur, ainsi que les pressions anthropiques, les événements liés à l'environnement, et tout autre entité interagissant avec l'écosystème, dans un même formalisme, celui des automates temporisés. Un avantage important de ce formalisme est de pouvoir représenter les contraintes temporelles ainsi que l'incertitude inhérente à l'évolution du stock d'une ressource. La discrétisation de la biomasse selon des niveaux usuels pour l'utilisateur permet une description générique de la topologie d'une espèce, ou d'une pression, qui autorise la construction automatique du réseau d'automates représentant l'écosystème. La génération automatique de l'automate, présentée dans cet article, repose sur un algorithme complexe qui nécessite des étapes de filtrage (regroupement des états intermédiaires, regroupement des transitions) afin de produire un automate suffisamment complet mais également exploitable dans la pratique. Les benchmarks ont montré qu'il était possible de traiter un écosystème de huit espèces et deux pressions de pêche, ce qui est généralement suffisant pour représenter un écosystème d'étude puisque nous ne nous intéressons qu'aux espèces pêchées ou observées (pour les écologues, le terme espèce correspond généralement à un groupe d'espèces, appelé groupe trophique). Une fois le modèle généré, des patrons de requêtes peuvent être utilisés pour explorer le modèle selon des scénarios prédictifs. Les patrons de requêtes proposés permettent une grande variété d'interrogations, puisqu'il est possible d'obtenir des informations sur les délais d'obtention d'une situation donnée, la possibilité d'atteindre une situation indésirable, les états possibles à une date donnée, la continuité d'une propriété au fil du temps, etc. Étant donné la modélisation choisie, les réponses aux requêtes sont généralement qualitatives (pour les états) et donc directement interprétables par

l'utilisateur qui est en général directement le décideur, dans notre cas le gestionnaire des pêches par exemple. Puisque le temps de génération de l'automate est faible (de l'ordre de quelques secondes), il lui est possible d'engager un processus interactif, en modifiant son scénario de pêche, en regénérant un nouveau modèle, puis en interrogeant de nouveau le modèle généré. À notre connaissance, EcoMata est un outil original permettant la modélisation qualitative d'un écosystème et son interrogation à l'aide de requêtes explicites qui produisent des résultats qualitatifs et sur de longues périodes de temps.

Les perspectives de ce travail sont de s'intéresser maintenant aux requêtes de type "Que faire pour ?". Nous étudions pour cela les approches de type synthèse de contrôleur (Altisen *et al.*, 2005) sur les automates temporisés pour les adapter à notre contexte. L'idée est par exemple de donner au logiciel la possibilité de proposer des politiques de pêches garantissant des états spécifiques du système, comme par exemple des états acceptables pour certaines espèces. D'un point de vue applicatif, nous effectuons une évaluation rétrospective, en travaillant sur une série de données concernant la Mer du Nord sur une période assez longue, afin de comparer les prédictions produites par EcoMata et les données réelles. Ceci nous permettra en particulier d'analyser finement l'impact sur les résultats de la plus ou moins grande précision du modèle, résultant de la plus ou moins grande simplification lors de la génération automatique.

Remerciements

Merci à Guy Fontenelle, Thomas Guyet et Yves-Marie Bozec pour leur collaboration.

8. Bibliographie

Ahmad J., Bourdon J., Eveillard D., Fromentin J., Roux O., Sinoquet C., « Temporal constraints of a gene regulatory network : Refining a qualitative simulation », *BioSystems*, vol. 98, n° 3, p. 149-159, 1991.

Altisen K., Bouyer P., Cachat T., Cassez F., Gardey G., « Introduction au contrôle des systèmes temps-réel », *European Journal of Automation*, vol. 39, n° 1-2-3, p. 367-380, 2005.

Alur R., Dill D., « A theory of timed automata », *Theoretical computer science*, vol. 126, p. pp. 183-235, 1994.

Batt G., Page M., Cantone I., Gessler G., Monteiro P., Jong H. D., « Efficient parameter search for qualitative models of regulatory networks using symbolic model-checking », *BioInformatics, ECCB10 Special Issue*, vol. 26, n° 18, p. i603-i610, 2010.

Beaujouan V., Durand P., Ruiz L., « Modelling the effect of the spatial distribution of agricultural practices on nitrogen fluxes in rural catchments », *Ecological Modelling*, vol. 137, n° 1, p. 93-105, 2001.

Berard B., Bidoit M., Finkel A., Laroussine F., Petit A., Schnoebelen P., McKenzie P., *Systems and Software Verification. Model-Checking Techniques and Tools*, Springer-Verlag, 2001.

Clarke E., Grumberg O., Peled D., *Model-Checking*, MIT Press, 2002.

Clarke E. M., Emerson E. A., Sistla A. P., « Automatic verification of finite-state concurrent systems using temporal logic specifications », *ACM Transactions on Programming Languages and Systems*, vol. 8, p. 244-263, 1986.

Cordier M.-O., Garcia F., Gascuel-Odoux C., Masson V., Salmon-Monviola J., Tortrat F., Trépos R., « A machine learning approach for evaluating the impact of land use and management practices on streamwater pollution by pesticides », *MODSIM'05 (International Congress on Modelling and Simulation)*, p. 2651-2657, 2005.

Guerrin F., Dumas J., « Knowledge representation and qualitative simulation of salmon redd functioning. Part I : qualitative modeling and simulation », *Biosystems*, vol. 59, n° 2, p. 75-84, 2001.

Henzinger T., Nicollin X., Sifakis J., Yovine S., « Symbolic model checking for real-time systems », *Information and Computation*, vol. 111, n° 2, p. 193-244, 1994.

Largouët C., Cordier M.-O., « Patrons de scenarios pour l'exploration qualitative d'un écosystème », *Actes de Reconnaissance des Formes et Intelligence Artificielle (RFIA'10)*, 2010.

Largouët C., Cordier M.-O., Bozec Y.-M., Fontenelle G., « Use of timed automata and model checking for exploring scenarios on ecosystem models », *Environmental Modelling and Software*, n.d. En cours de révision.

Larsen K., Pettersson P., Yi W., « UPPAAL in a Nutshell », *Journal of Software Tools for Technology Transfer*, vol. 1, n° 1-2, p. 134-152, 1997.

Murray J. D., *Mathematical Biology I : An Introduction*, vol. 17 of *Interdisciplinary Applied Mathematics*, Springer New York, 2002.

Rickel J., Porter B., « Automated Modeling of Complex Systems to Answer Prediction Questions », *Artificial Intelligence Journal*, vol. 93, p. 201-260, 1997.

Rykiel J., « Artificial Intelligence and Expert Systems in Ecology and Natural Resources Management », *Ecological Modelling*, vol. 46, n° 1-2, p. 3-8, 1989.

Salles P., Bredeweg B., Araújo S., « Qualitative models about stream ecosystem recovery : Exploratory studies », *Ecological Modelling*, vol. 194, n° 1-3, p. 80-89, 2006.

Shults B., Kuipers B., « Proving properties of continuous systems : qualitative simulation and temporal logic », *Artificial Intelligence*, vol. 92, n° 1-2, p. 91-129, 1997.

Tulllos D., Neumann M., « A Qualitative Model for Characterizing Effects of Anthropogenic Activities on Benthic Communities », *Ecological Modeling*, vol. 196, p. 209-220, 2006.

Veresi A., Toffolatti S., Zocchi G., Guglielmann R., Ironi L., « A new approach to modelling the dynamics of oospore germination in Plasmopara viticola », *European Journal of Plant Pathology*, vol. 128, n° 18, p. 113-126, 2010.

Yovine S., « Kronos : A Verification Tool for Real-Time Systems (Kronos User's Manual Release 2.2) », *International Journal of Software Tools for Technology Transfer*, vol. 1, p. 123-133, 1997.

Yovine S., « Model Checking Timed Automata », *Embedded Systems*, LNCS Springer-Verlag, 1998.

ANNEXE POUR LE SERVICE FABRICATION
A FOURNIR PAR LES AUTEURS AVEC UN EXEMPLAIRE PAPIER
DE LEUR ARTICLE ET LE COPYRIGHT SIGNE PAR COURRIER
LE FICHIER PDF CORRESPONDANT SERA ENVOYE PAR E-MAIL

1. ARTICLE POUR LA REVUE :
RSTI - ISI – 16/2011. SI pour l'environnement

2. AUTEURS :
Yulong Zhao[1,3] — Christine Largouët[1,2] — Marie-Odile Cordier[1,3]

3. TITRE DE L'ARTICLE :
EcoMata : Un logiciel d'aide à la décision pour améliorer la gestion des écosystèmes

4. TITRE ABRÉGÉ POUR LE HAUT DE PAGE MOINS DE 40 SIGNES :
EcoMata

5. DATE DE CETTE VERSION :
24 novembre 2011

6. COORDONNÉES DES AUTEURS :

– adresse postale :

[1] IRISA, Campus universitaire de Beaulieu, 35042 Rennes, France

[2] AGROCAMPUS OUEST, 65 rue de St-Brieuc, 35042 Rennes, France

[3] Université de Rennes 1, Campus universitaire de Beaulieu 35042 Rennes, France

contact : yulong.zhao@irisa.fr, largouet@agrocampus-ouest.fr, cordier@irisa.fr

– téléphone : 02 99 84 74 82

– télécopie :

– e-mail : yulong.zhao@irisa.fr, largouet@agrocampus-ouest.fr, cordier@irisa.fr

7. LOGICIEL UTILISÉ POUR LA PRÉPARATION DE CET ARTICLE :
LaTeX, avec le fichier de style `article-hermes.cls`,
version 1.23 du 17/11/2005.

8. FORMULAIRE DE COPYRIGHT :
Retourner le formulaire de copyright signé par les auteurs, téléchargé sur :
`http://www.revuesonline.com`

SERVICE ÉDITORIAL – HERMES-LAVOISIER
14 rue de Provigny, F-94236 Cachan cedex
Tél. : 01-47-40-67-67
E-mail : revues@lavoisier.fr
Serveur web : http://www.revuesonline.com