HAL
open science

Vérification du simulateur PARIS et préparation à sa validation

Benjamin Astre, Claude Mazel, René Baumont, Pascal Carrère, Laurent Pérochon

To cite this version:

Benjamin Astre, Claude Mazel, René Baumont, Pascal Carrère, Laurent Pérochon. Vérification du simulateur PARIS et préparation à sa validation. Sciences de l'environnement. 2004. hal-03326227

HAL Id: hal-03326227

https://hal.inrae.fr/hal-03326227

Submitted on 25 Aug 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT SUPERIEUR
D'INFORMATIQUE
DE MODELISATION
ET DE LEURS APPLICATIONS

COMPLEXE DES CEZEAUX
BP 125 – 63173 AUBIERE CEDEX

INSTITUT NATIONAL
DE LA RECHERCHE
AGRONOMIQUE

UNITE DE RECHERCHES SUR LES
HERBIVORES
THEIX
63122 ST-GENES-CHAMPANELLE

Rapport de Stage 2ème année F4

Vérification du simulateur PARIS et préparation à sa validation

Tome I

Présenté par : Benjamin ASTRE
Responsable ISIMA : M Claude MAZEL
Responsable INRA : M René BAUMONT
M Pascal CARRERE
M. Laurent PEROCHON

Du 5 Avril au 23 Septembre

Remerciements

Je tiens à remercier M. René Baumont et M. Pascal Carrère pour leurs expertises des résultats du simulateur et pour l'attention qu'ils m'ont consacré tout au long du stage. Je remercie aussi M. Laurent Pérochon pour ses conseils d'analyse du programme.

Je remercie également M. Claude Mazel et Mme Christine Force pour leurs conseils pour la validation.

Enfin merci à toute l'équipe RAPA de l'URH à l'INRA de Clermont-Ferrand/Theix pour son accueil.

Table des abréviations

FGEP : Fonctionnement et Gestion de l'Ecosystème Prairial
FMI : Fonction de motivation à ingérer
FSAT : Fonction de satiété
IN : Indice de Nutrition
INRA : Institut National de la Recherche Agronomique
ISIMA : Institut Supérieur d'Informatique de Modélisation et de leurs Applications
LIMOS : Laboratoire d'Informatique de Modélisation et d'optimisation des systèmes
MS : Matière Sèche
RAPA : Relations Animal-Plantes et Alimentation
RS : compartiment Reproductif Sec
RV : compartiment Reproductif Vert
SLA : Specific Leaf Area, surface de feuille (cm^2) par unité de biomasse (g)
UML : Unified Modeling Language
UP : Unified Process
URH : Unité de Recherche sur les Herbivores
VS : compartiment Végétatif Sec
VV : compartiment Végétatif Vert

Glossaire

Biomasse : quantité de matière disponible sur une cellule ($g\ m^{-2}$).
C++ : Langage de programmation orienté objet, dérivé du langage C
Défolier : manger, brouter, prélever de la biomasse.
Grégarisme : relation sociale au sein d'un troupeau de ruminants qui pousse ces animaux à rester grouper.
Leadership : relation sociale poussant les animaux à suivre un animal particulier du troupeau (le leader) pour les grands déplacements sur la parcelle.
Ruminant : mammifère pourvu d'un système digestif propre à la rumination.
Sénescence : vieillissement naturel des tissus et de l'organisme.

Table des figures et illustrations

FIGURE 1-1 : INTERACTIONS ENTRE LES MODELES 11
FIGURE 1-2 : AUTOMATE PAR HIERARCHIE DES ACTIONS, UTILISE JUSQU'AU STAGE DE M. JOUVEN 13
FIGURE 1-3 : EXEMPLE DE PROBABILITES POUR L'AUTOMATE DECISIONNEL 13
FIGURE 1-4 : REPRESENTATION DU CHAMP DE VISION D'UN RUMINANT 14
FIGURE 1-5 : DIAGRAMME DE CLASSE DU MODELE VEGETAL 16
FIGURE 1-6 : DIAGRAMME DE CLASSE DU MODELE ANIMAL 17
FIGURE 1-7 : DIAGRAMME DE CLASSE DE L'INTERCONNEXION ENTRE MODELES 18
FIGURE 1-8 : DIAGRAMME DE CLASSE DE L'ECHEANCIER 18
FIGURE 3-1 : SCHEMA DU NOUVEL AUTOMATE DECISIONNEL 25
FIGURE 3-2 : DIAGRAMME D'ACTIVITE POUR LE NOUVEL AUTOMATE DECISIONNEL 26
FIGURE 4-1 : LES DEUX COURBES EXTREMES DE LA VITESSE D'INGESTION EN FONCTION DE LA MASSE DE LIMBES VERTS 30
FIGURE 4-2 : EVOLUTION DE GAMMAVV EN FONCTION DE LA SOMME DES TEMPERATURES 31
FIGURE 4-3 : NOUVEAU PARAMETRAGE DE LA FONCTION DE MONTAISON 33
FIGURE 4-4 : EVOLUTION DES BIOMASSES DES COMPARTIMENTS AU COURS D'UNE ANNEE AVANT LE CALIBRAGE (A GAUCHE) ET APRES (A DROITE) 34
FIGURE 5-1 : COMPARAISON DES RESULTATS ENTRE 1 REPLICATION ET 5 REPLICATIONS 38
FIGURE 5-2 : SIMULATION AVEC LA FONCTION DE DISTANCE « MIN » 39
FIGURE 5-3 : POSITIONNEMENTS NON GREGAIRES POSSIBLES AVEC LA FONCTION « MIN » 40
FIGURE 5-4 : SIMULATION AVEC LA FONCTION DE DISTANCE « BARYCENTRE » PAR RAPPORT A TOUT LE TROUPEAU AVEC LES MEMES POSITIONS DE DEPART QUE LA SIMULATION AVEC LA FONCTION « MIN » (FIGURE 5-2) 40
FIGURE 5-5 : EVOLUTION DES BIOMASSES POUR LES QUATRE CHARGEMENTS 66.7, 33.3, 16.7 ET 8.3 BREBIS/HA 43
FIGURE 5-6 : EVOLUTION DES BIOMASSES DE LA PARCELLE PENDANT LA SAISON ETUDIEE ICI 44
FIGURE 5-7 : EVOLUTION DE LA MOYENNE DES QUANTITES INGEREES PAR JOUR DE CHAQUE ANIMAL, DE LEURS DUREES D'INGESTION ET DE LEURS VITESSE D'INGESTION 45
FIGURE 5-8 : EVOLUTION DE LA BIOMASSE DU COMPARTIMENT RV POUR LA DEUXIEME METHODE (A GAUCHE) ET LA TROISIEME (A DROITE) 46
FIGURE 5-9 : EVOLUTION DE LA VARIABILITE DE LA BIOMASSE DU COMPARTIMENT RV ENTRE LES CELLULES DE LA METHODE 1 (A GAUCHE) ET LA METHODE 2 (A DROITE) 47

Résumé

Ce stage s'inscrit dans un programme de recherche pluri-disciplinaire sur « l'utilisation par le pâturage et l'évolution de la végétation des surfaces herbagères hétérogènes et peu chargées ». Dans ce cadre, un **simulateur** a été développé à des fins de connaissance, pour fournir un outil permettant de simuler le fonctionnement d'un système parcelle troupeau selon les caractéristiques des animaux, de la végétation et les paramètres de gestion du système. Il a été conçu sur le principe des **multi-agents** afin de gérer chaque animal du troupeau indépendamment, et tenir compte de la variabilité comportementale.

Le développement du simulateur s'effectue par prototypes successifs. J'ai travaillé sur un prototype associant les modèles de la végétation, de l'animal et de la cohésion sociale au sein du groupe. L'objectif du stage était de réaliser la **vérification** informatique de ce prototype, puis d'entamer une **validation** des résultats dans le but de garantir sa fiabilité et la correspondance entre les résultats qu'il fournit et les données expérimentales mises à ma disposition.

Mon premier travail a consisté à améliorer la gestion des fichiers de sortie du simulateur afin d'en faciliter l'analyse. Puis j'ai vérifié le fonctionnement du simulateur et j'ai corrigé les erreurs qui résultaient du développement de nouvelles fonctions comme la cohésion sociale et l'automate décisionnel pour gérer les choix d'activité des animaux. J'ai ensuite implémenté et vérifié des améliorations du fonctionnement biologique des modèles de la végétation et de l'animal. Cela a ensuite nécessité de refaire en partie l'étape de calibrage de certains paramètres des modèles. Enfin, j'ai appliqué un protocole d'expérimentation informatique faisant varier le niveau d'exploitation de la parcelle pour vérifier la cohérence du fonctionnement biologique du prototype.

Le simulateur est implémenté en **C++** sous **LINUX**. J'ai ainsi travaillé sous la distribution Linux Mandrake 10.0 pour tous les traitements effectués sur le code du simulateur et pour l'exécution des simulations. Les sorties du simulateur ont ensuite été traitées et analysées sous « Windows » avec le tableur « **Excel ».**

Ainsi à l'issue du stage j'ai obtenu un prototype fiabilisé dans son fonctionnement avec un animal unique. Les défauts qui avaient été soulevés lors du précédent stage de validation ont été rectifiés. Les premières analyses effectuées pour la validation du simulateur avec plusieurs animaux ont permis de confirmer que le fonctionnement biologique du système modélisé est respecté. Mais elles m'ont également permis de mettre en évidence des disfonctionnements liés à la gestion du troupeau. Des pistes d'amélioration sont proposées.

Mots-clés : simulateur, multi-agents, vérification, validation, C++, LINUX, Excel

Abstract

This training period contributes to a multidisciplinary research program on "the use by grazing and the evolution of heterogeneous grasslands". Within this context, a **simulator** was developed for knowledge ends, to provide a tool that allows simulating a herd-grassland system in relation with animal and vegetation characteristics, and management of the system. It was conceived on the principle of **multi-agents** in order to manage each animal of the herd independently, and taking account of behavioural variability.

The development of the simulator is made with successive prototypes. I worked on a prototype associating the vegetation model, the animal model and social cohesion within the group. The objective of my training period was to make the data-processing **verification** of this prototype, then to start a **validation** of results in order to guarantee its reliability and the correspondence between the results it provides and experimental data.

My first work consisted in improving management of output files of the simulator in order to facilitate their analysis. Then I verified how the simulator operates and I corrected errors which resulted from the development of new functions like social cohesion and decisional automate to manage choices of animal's activity. After that, I implemented and verified improvements in the biological models of vegetation and animal. That required to partly recalibrating the models parameters. Lastly, I used the simulator to test the influence of the plot exploitation level in order to verify the coherence of the biological working of the prototype.

The simulator is implemented in **C++** language under **LINUX** system. Thus I worked under the distribution Linux Mandrake 10.0 for all treatments made on the simulator code and for simulations executions. Then I analysed simulator outputs under "Windows" with the spreadsheet "**Excel".**

At the end of my training period I obtained a reliable prototype with just one animal. The defects which were pointed out at the end of the previous validation training period were rectified. The first analyses of simulator validation with several animals confirm that the biological working of the modelled system is respected. But they also allowed me to highlight bugs related to herd management. Some tracks of improvement are proposed.

Key words: simulator, multi-agents, verification, validation, C++, LINUX, Excel

Table des matières

TOME I

REMERCIEMENTS

TABLE DES ABREVIATIONS

GLOSSAIRE

TABLE DES FIGURES ET ILLUSTRATIONS

RESUME

ABSTRACT

TABLE DES MATIERES

INTRODUCTION **9**

1 DESCRIPTION DU SIMULATEUR **10**

1.1 LES OBJECTIFS DU SIMULATEUR 10
1.2 DESCRIPTION DU SIMULATEUR 10
1.3 DESCRIPTION DES MODELES 11
1.4 ETAT DE L'IMPLEMENTATION DU SIMULATEUR : PROTOTYPE 2003 16
1.5 PARTIES VALIDEES 19
1.6 OBJECTIFS DU STAGE 19

2 AMELIORATION DE LA GESTION DES FICHIERS DE SORTIE **20**

2.1 LES MODIFICATIONS APPORTEES 20
2.2 TRAVAIL RESTANT A FAIRE 23

3 VERIFICATION DU « PROTOTYPE 2003 » **24**

3.1 VERIFICATION DE L'AUTOMATE DECISIONNEL 25
3.2 AUTRES CORRECTIONS APPORTEES 27

4 AMELIORATIONS APPORTEES AUX MODELES BIOLOGIQUES **29**

4.1 DESCRIPTION DES AMELIORATIONS APPORTEES 29
4.2 CALIBRATION 31

5 VALIDATION **36**

5.1 QU'EST CE QUE LA VALIDATION : CONCEPTS GENERAUX ET TECHNIQUES 36
5.2 OUTILS SPECIFIQUES DEVELOPPES POUR LA VALIDATION 36
5.3 SIMULATIONS REALISEES 37

6 DISCUSSION **48**

6.1 CONTRAINTES TECHNIQUES 48
6.2 FORCES ET FAIBLESSES DU SIMULATEUR 48
6.3 L'AVENIR DU SIMULATEUR 49

CONCLUSION **50**

REFERENCES BIBLIOGRAPHIQUES **51**

TOME II

TABLE DES FIGURES ET ILLUSTRATIONS ..... **II**

TABLE DES MATIERES ..... **III**

ANNEXE 1 : DEPENDANCES ET LIEU DE CREATION DES SORTIES DU SIMULATEUR ..... **IV**

ANNEXE 2 : GUIDE DE FONCTIONNEMENT DE DOXYGEN ..... **VII**

ANNEXE 3 : EVOLUTION DES TEMPERATURES ET DU RAYONNEMENT SOLAIRE AVEC LE NOUVEAU FICHIER ENVIRONNEMENT STANDARD ..... **VIII**

ANNEXE 4 : VALEUR DES PARAMETRES DU FICHIER *FACIES.CSV* STANDARD UTILISE DANS LES SIMULATIONS ..... **X**

ANNEXE 5 : QU'EST CE QUE LA VALIDATION, CONCEPTS GENERAUX ET TECHNIQUES .. **XII**

ANNEXE 6 : DESCRIPTION DES PARAMETRES D'ENTREE DU SIMULATEUR ..... **XXV**

ANNEXE 7 : DESCRIPTION DES FICHIERS DE SORTIE DU SIMULATEUR A L'ISSUE DU STAGE ..... **XXVIII**

ANNEXE 8 : RESULTATS DE LA VARIATION DE CHARGEMENT A SURFACE CONSTANTE XXX

Introduction

Mon stage qui s'est déroulé du 5 avril au 23 septembre 2004 à l'Institut National de la Recherche Agronomique de Clermont-Ferrand – Theix, s'inscrit dans le cadre de l'élaboration du simulateur parcelle/troupeau PARIS (PAsture Ruminant Interactions Simulator) dont la conceptualisation a été réalisée par les équipes d'accueil : l'équipe RAPA (Relation animal plante et aliments) de l'Unité de Recherche sur les Herbivores et l'équipe FGEP (Fonctionnement et gestion de l'écosystème prairial) de l'Unité d'Agronomie, en collaboration avec le LIMOS. Les élèves ingénieurs de l'ISIMA ont participé à l'implémentation du simulateur durant les stages précédents.

Ce simulateur a été développé dans le cadre d'un projet de recherche intitulé « *l'utilisation par le pâturage et l'évolution de la végétation des surfaces herbagères hétérogènes et peu chargées* » dans le but d'étudier les interactions entre un troupeau d'herbivores et une surface prairiale, et d'analyser l'utilisation d'une ressource fourragère plus ou moins complexe par les animaux. Les finalités de ce travail porteront autant sur des améliorations des pratiques de gestion (valeur d'usage de parcelles hétérogènes pour nourrir les animaux), que sur des recommandations en terme d'impact environnemental (limitation de l'embroussaillement des parcelles, maintien de la diversité biologique).

Après une phase de conception et d'implémentation, le projet a entamé une phase de vérification, afin de s'assurer de la conformité du simulateur avec les objectifs affichés et sa cohérence avec le système modélisé. Ce travail a été réalisé pour un premier prototype du simulateur fonctionnant avec un seul animal lors du stage de Magali Jouven en 2003 [6]. L'objectif de mon travail était, après avoir réalisé une étude bibliographique sur les méthodes de vérification et de validation, de choisir les techniques les mieux adaptées, et de les appliquer afin de vérifier le simulateur pour un fonctionnement avec plusieurs animaux, et de lancer les amorces de la phase de validation biologique.

Je commencerai dans ce rapport par présenter l'état du simulateur sur lequel j'ai travaillé. Ensuite je détaillerai le travail de vérification que j'ai effectué avec ces conséquences en terme de changements conceptuels du simulateur. Enfin je décrirai les premières étapes du travail de validation que j'ai réalisé.

1 Description du simulateur

1.1 Les objectifs du simulateur

Ce simulateur reproduit les interactions entre un troupeau d'herbivores (ruminants) et la parcelle sur laquelle il pâture. Et cela afin de comprendre les mécanismes de fonctionnement de cette interaction dans le processus de pâturage. On pourra ainsi évaluer l'impact de la gestion d'une parcelle pour les animaux (comportement, ingestion) et pour la végétation (évolution en quantité et en qualité). Ce modèle permettra de progresser dans la compréhension des mécanismes qui déterminent l'utilisation d'une parcelle par un troupeau de ruminants. En ce sens il s'agit de concevoir un modèle heuristique, qui sera un outil de recherche complémentaire à l'expérimentation dans les travaux de recherches sur la compréhension de l'utilisation d'une parcelle par un troupeau de ruminants.

A terme, ce simulateur pourra aider à concevoir un outil d'aide à la décision, qui permettra de tester différentes situations de gestion et fera apparaître les itinéraires les plus souhaitables en fonction des objectifs de l'éleveur (maintien de la production, conservation de la ressource, limitation de l'embroussaillement, accroissement de la diversité). Il pourra à terme fournir les bases pour évaluer et choisir les stratégies de gestion des parcelles pour les éleveurs.

1.2 Description du simulateur

Ce simulateur est basé sur un système multi-agents, les agents étant ici les animaux. Un tel système consiste à faire coopérer un ensemble d'entités ou agents dotés d'un comportement intelligent, coordonner leurs buts et leurs plans d'actions pour résoudre un problème [22]. Un agent est une entité physique ou abstraite caractérisée par son autonomie dans la prise de décision, ses connaissances sur lui-même et les autres, et sa capacité d'agir. L'approche multi-agents a été retenue parce qu'elle s'adapte remarquablement bien aux modèles nécessitant la prise en compte de l'espace et des comportements individuels [23].

De l'analyse du système prairie troupeau il est apparu qu'il pouvait s'organiser en trois sous-ensembles, suffisamment indépendants pour être conçus séparément. Le premier modèle est dédié au fonctionnement de la végétation, le deuxième au fonctionnement de l'animal et le troisième au comportement social et spatial du troupeau. Ainsi le modèle Végétal (V) gère l'état de la végétation sur la parcelle, c'est-à-dire l'évolution de la biomasse végétale au cours du temps. Le modèle Animal (A) s'occupe du comportement d'ingestion d'un animal, il gère ses activités de pâturage incluant ses déplacements à courte distance et son métabolisme. Et enfin le modèle Social et Spatial (S) traite les actions concernant le troupeau dans son ensemble, à travers les mécanismes de comportement sociaux et leurs implications spatiales (déplacements long, mémoire).

Ces modèles communiquent entre eux pour former le simulateur, qui peut être en ce sens identifié à un méta-modèle (Figure 1-1). On identifie trois communications :

- Entre les modèles V et A, V informe A des nouvelles quantités de biomasse disponibles pour la défoliation, et A informe V des quantités défoliées.
- Entre les modèles A et S, A fait appel à S à chaque fois qu'un animal a choisi une action impliquant un mouvement de l'ensemble du troupeau.
- Entre les modèles S et V, via les conséquences des déplacements longs ou de la mémoire sur l'utilisation différentielle des sites alimentaires de la parcelle.

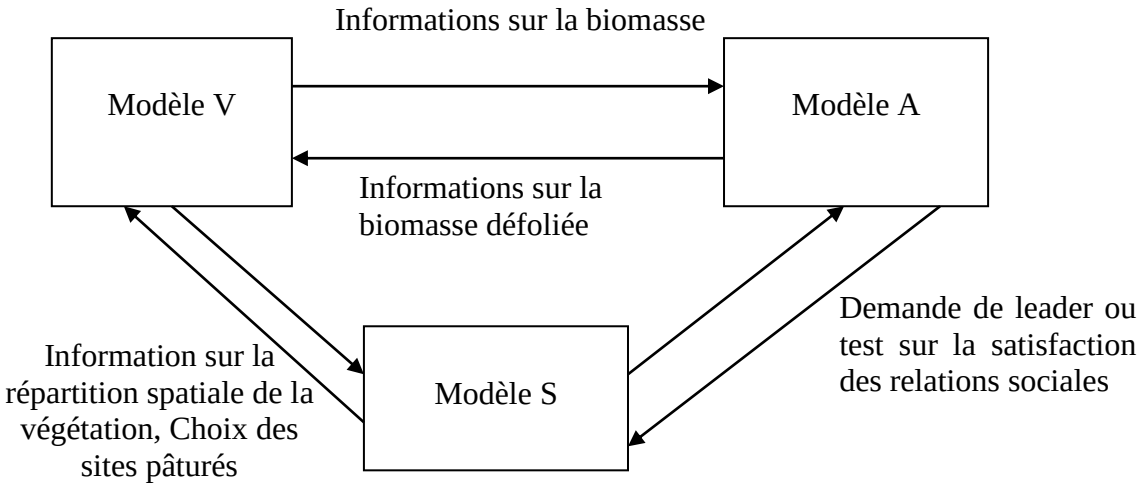

Figure 1-1 : Interactions entre les modèles

Ces trois modèles sont détaillés dans les paragraphes suivants.

1.3 Description des modèles

1.3.1 Modèle végétal

1.3.1.1 Modélisation de la parcelle

L'unité spatiale la plus intégrative du simulateur est la parcelle qui représente la surface de terrain sur laquelle pâture les ruminants. Afin de pouvoir simuler la variabilité existante dans une parcelle en conditions réelles, il a été fait le choix de découper la surface totale en unités élémentaires appelées cellules. Il a été choisi de réaliser des cellules de forme hexagonale car cela permet d'avoir une équidistance entre les centres des cellules voisines. La surface de la cellule a été définie de façon à ce qu'elle corresponde à la surface sur laquelle le ruminant peut défolier sans bouger ses pattes avant (station alimentaire, soit environ 0.1 m² pour un ovin). Une hypothèse forte du modèle est que tous les évènements intervenant sur une cellule le font de manière homogène

Chaque cellule est constituée de quatre compartiments qui représentent les états possibles de la végétation (végétatif ou reproductif ; vert ou sec). Il y en a quatre distincts : végétatif vert (VV) reproductif vert (RV), végétatif sec (VS) et reproductif sec (RS). La végétation est décrite par des communautés végétales appelées faciès. Les cellules adjacentes appartenant à un même faciès végétal définissent un site végétal. Ainsi on peut définir deux niveaux de segmentation de la parcelle : par faciès et sites végétal.

1.3.1.2 Traitement effectué

Le modèle V correspond à un simulateur (il peut fonctionner sans les autres modèles), en effet il simule l'évolution de la végétation de la parcelle au cours du temps. Ainsi la seule interaction avec le reste du simulateur est l'information donnée au modèle animal sur l'état de la végétation en un lieu donné de la parcelle, et en retour l'acquisition des quantités de biomasse défoliée par le troupeau qui doivent être retirées des cellules. Ainsi ce modèle peut très bien se suffire à lui-même pour simuler l'évolution de la végétation d'une parcelle sans

troupeau, cela présente l'avantage de permettre une vérification du modèle végétal dans son fonctionnement sans animaux avant de passer au simulateur complet.

L'évolution de la végétation est un mécanisme continu qui a été discrétisé à des fins de simplification de la modélisation. Le choix a été de mettre à jour les informations sur l'état de la cellule après chaque défoliation en ce qui concerne le prélèvement de biomasse (donc conditionné au fonctionnement du module animal) et en fin de journée pour ce qui est du fonctionnement propre de la végétation (croissance, sénescence). Quatre fonctions principales gèrent le fonctionnement de la végétation :

- *Fonction Croissance* : elle provoque l'augmentation de la biomasse du VV due à la pousse de la végétation pendant la journée.
- *Fonction Montaison* : elle permet le développement des parties de reproduction des végétaux en faisant passer de la biomasse de VV à RV
- *Fonction Sénescence* : elle gère le vieillissement de la végétation en transférant de la biomasse de VV à VS et de RV à RS.
- *Fonction Litière* : elle modélise la disparition de la biomasse du système. Elle enlève donc de la biomasse aux compartiments VS et RS.

1.3.2 Modèle animal

Dans ce modèle, on gère l'état d'un animal, c'est-à-dire sont état physiologique (sur le plan de l'ingestion et de la digestion) d'une part et les actions qu'il entreprend et leurs conséquences d'autre part. Cela exclut les actions nécessitant de longs déplacements sur la parcelle qui ne peuvent être gérées qu'avec les relations sociales au sein du troupeau (gérées dans le modèle S, voir 1.3.3).

Le ruminant peut entreprendre plusieurs actions : Manger (défolier), boire, se reposer sur l'aire de couchage (repos long) ou rester inactif sur place (repos court). Cette dernière action correspond à l'absence de besoin d'effectuer l'une des trois actions précédentes. Le choix de la prochaine activité de l'animal est réalisé grâce à un automate décisionnel (voir 1.3.2.1). On peut remarquer que les actions « boire » et « repos long » ne se font qu'à certains endroits spécifiques de la parcelle (il n'y a en général qu'un seul réservoir d'eau sur une parcelle, et l'animal choisit une aire de couchage principale). Ainsi ces deux actions risquent de nécessiter un déplacement long et le cas échéant, le modèle A fera appel au modèle S.

Il existe une autre action que le ruminant peut entreprendre, il s'agit de la rumination. Cette action est un peu particulière puisqu'elle ne peut se faire que si l'animal est au repos (court ou long).

1.3.2.1 L'automate décisionnel

Deux automates ont été étudiés pour déterminer l'action que l'animal souhaite réaliser. Le premier très simplifié (Figure 1-2) se basait seulement sur une hiérarchie des actions. Ainsi si l'animal voulait défolier ou boire, il mangeait et ne buvait que lorsqu'il n'avait plus envie de manger. Cela était dû au choix de prendre une priorité plus importante pour défolier que pour boire dans la hiérarchie des actions.

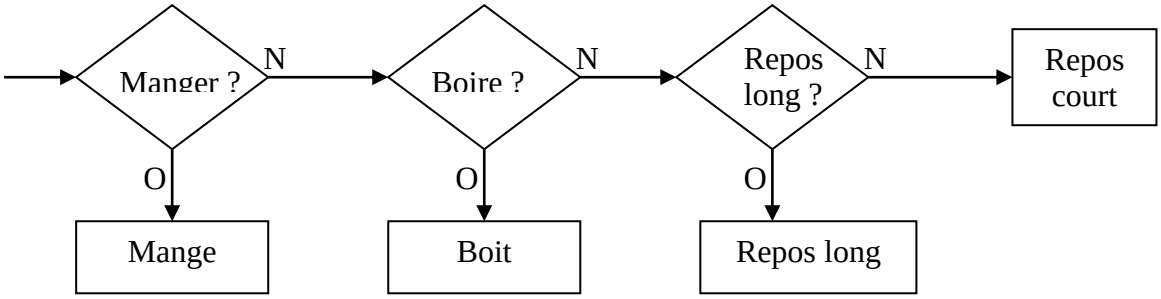

Figure 1-2 : Automate par hiérarchie des actions, utilisé jusqu'au stage de M. Jouven

Il a été complété (par S. Portal) pour déboucher sur un second automate décisionnel plus complet. On calcule tout d'abord des priorités pour chaque action (défolier, boire et se reposer) à partir des paramètres métaboliques de l'animal. Ensuite on ramène ces priorités entre 0 et 1, tout en laissant la possibilité d'avoir un repos court (période d'inactivité) grâce à un paramètre P, afin d'obtenir une probabilité de réalisation pour chacune des quatre actions. Ensuite on répartit ces probabilités sur le segment [0 ; 1] (Figure 1-3) afin d'effectuer un tirage aléatoire entre 0 et 1 pour déterminer l'action choisie.
Soit :

m = priorité de l'action manger
b = priorité de l'action boire
r = priorité de l'action repos long
P = le paramètre pour le repos court >= 1
Si **T** = (m+b+d)*P
Alors **Pm** = probabilité de choisir l'action manger = **m/T**
Pb = probabilité de choisir l'action boire = **b/T**
Prl = probabilité de choisir l'action repos long = **r/T**
Prc = Probabilité de choisir de faire un repos court = **1-(1/P)**

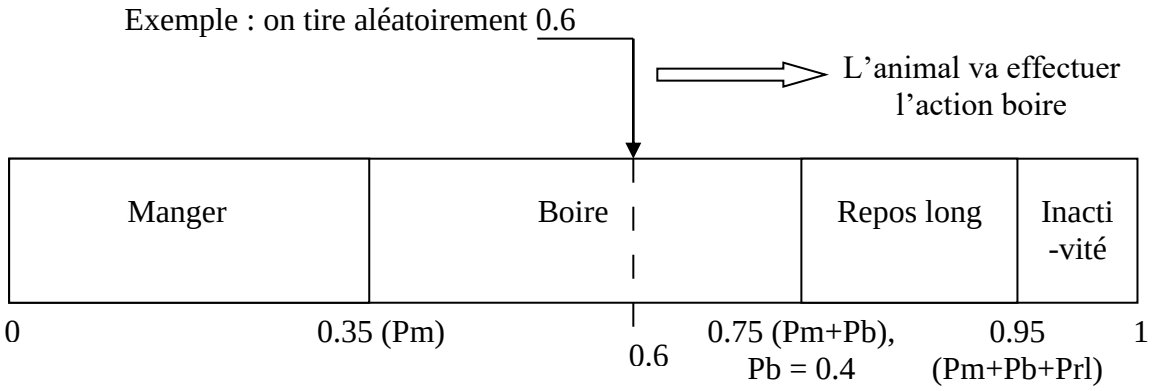

Figure 1-3 : Exemple de probabilités pour l'automate décisionnel

Cet automate permet plus de souplesse dans le choix d'une action, laissant toujours une chance à l'animal de faire n'importe laquelle des actions. Il a été implémenté lors du stage de S. Portal [5].

1.3.2.2 Description de l'action défolier

Lorsque l'animal choisit de manger, il va défolier une succession de cellules de végétation au cours de la séquence d'ingestion. Pour ce qui est de cette action, l'animal a deux possibilités :

- Se contenter des cellules de son champ de vision
- Vouloir changer d'endroit afin de trouver une végétation plus attractive ailleurs (cette deuxième possibilité étant impossible dans le prototype sur lequel j'ai travaillé car la mémoire n'était pas encore implémentée)

Dans le premier cas, l'animal va choisir une nouvelle cellule à défolier dans son champ de vision. Pour la choisir le ruminant se sert de la composition végétale des cellules, telle qu'il peut la percevoir. Le champ de vision de l'animal est défini par un angle de 120 degrés, et limité à trois rangées de cellule (distance par rapport à la tête de l'animal en terme de cellule), car l'animal se déplace tête baissée puisqu'il est en train de défolier. Ainsi ses possibilités de choix se limitent à 15 cellules (Figure 1-4).

Dans le second cas, il faut que l'animal soit le leader pour engendrer le mouvement du troupeau vers une nouvelle position, on fait donc appel ici au modèle S (voir 1.3.3).

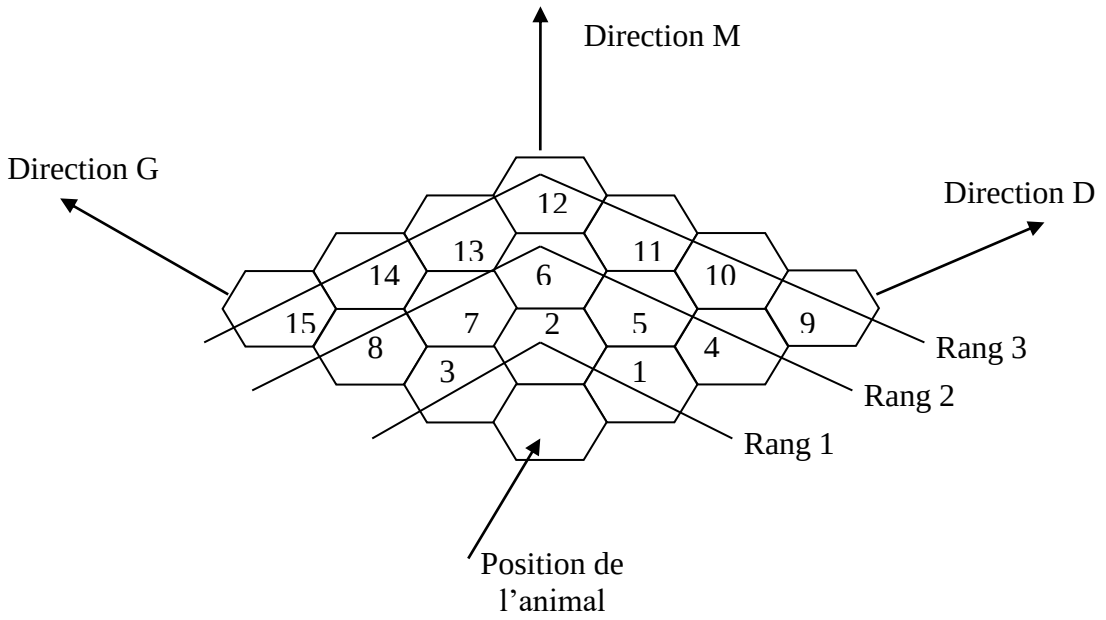

Figure 1-4 : Représentation du champ de vision d'un ruminant

Le choix de la cellule de destination se fera par un tirage aléatoire (ce qui introduit une part de stochasticité dans la défoliation). Pour cela nous devons associer une probabilité d'être choisie pour chacune des cellules du champ de vision. Elle s'obtient en combinant une probabilité liée à la position spatiale de la cellule dans le champ de vision et une probabilité liée à l'intérêt alimentaire de la cellule perçu par l'animal. La probabilité spatiale est obtenue en combinant une valeur liée à la direction (plus forte probabilité d'aller en face que sur les cotés) et une valeur liée au positionnement de la cellule (plus forte probabilité d'allers près que d'aller loin).

Les probabilités liées à la qualité des cellules perçue par l'animal sont obtenues en prenant les valeurs fournies par le modèle V et en appliquant l'une des trois fonctions de perception suivantes :

- Preference 1 : effectue un calcul empirique sur les proportions de chaque compartiment
- Preference 2 : calcul avec les préférences des animaux, basée sur des données expérimentales
- Optimal Foraging Theory : calcul selon la vitesse d'ingestion permise par la cellule

Enfin les probabilités finales sont obtenues par la formule suivante :

$$P_{finale} = P_{qualité}^{\mathbf{a}} * P_{direction/rang}^{\mathbf{b}}$$

Où **a** et **b** sont deux paramètres permettant d'ajuster l'influence de la perception et de la position d'une cellule sur les probabilités de choix des cellules par l'animal. C'est-à-dire, savoir quelle est la probabilité prépondérante entre la perception de l'animal de la qualité de la cellule et la position de la cellule dans le champ de vision. Ces deux paramètres ont été calibrés à 5 pour **a** et 0.3 pour **b** dans le cas de la fonction de perception Optimal Foraging Theory et à 15 et 0.3 pour Preference 1 (M. Jouven [6]).

Enfin on utilise le paramètre S (sélectivité) pour réduire le nombre de cellules pour lesquelles on va faire le tirage aléatoire. En effet, c'est un nombre compris entre 1 et 15 qui désigne les S meilleurs cellules (sur leurs probabilités) parmi lesquelles on doit tirer la cellule à défolier. Ce paramètre a été calibré lors du stage de M. Jouven [6] à la valeur de 10.

1.3.2.3 Evolution du métabolisme de l'animal

En seconde fonction le modèle A, fait évoluer l'organisme de l'animal en fonction des actions de ce dernier, en particulier, l'ingestion, la rumination et la prise d'eau (voir rapport de stage de F. Guerry [2]). Un modèle simple de la digestion et du métabolisme de l'animal (Sauvant et al [24] ; Baumont et al [25]) permet de simuler les variations de besoins alimentaires de l'animal qui se traduisent en état de motivation à ingérer ou de rassasiement, en besoin de prise d'eau ou de repos long. Ainsi ce modèle va contribuer au choix d'activité de l'animal. Ainsi l'élément primordial de cette évolution est la vitesse d'ingestion qui va permettre également d'influencer les choix de l'animal (voir Optimal Foraging Theory 1.3.2.2).

1.3.3 Modèle social

Ce modèle permet de simuler le comportement du troupeau de ruminants avec les relations de grégarité et de leadership existant entre ces ruminants (au lieu d'un seul animal (représentatif de l'animal moyen) comme cela était le cas dans les prototypes utilisés avant le stage de S. Portal [5]). Il a deux fonctions, la gestion des relations sociales au sein du troupeau et la gestion des déplacements longs sur la parcelle. Ces deux fonctions sont étroitement liées puisque ces déplacements ne peuvent se faire que grâce à la présence d'un leader.

Le grégarisme est la première relation sociale régissant un troupeau de ruminants. En effet, ces derniers forment un groupe indissociable, c'est-à-dire que chaque animal ne s'éloignera jamais beaucoup du reste du troupeau. On a ainsi introduit dans le simulateur un

test d'éloignement vérifiant si l'animal n'est pas trop loin du troupeau et par la même occasion s'il n'est pas trop proche des autres individus.

Le leadership constitue la deuxième relation sociale au sein du troupeau. Elle intervient lorsqu'un animal désire effectuer un déplacement long, ce désir est présent quand il veut manger, par exemple, mais ne trouve aucune cellule à son goût dans son champ de vision et souhaite donc aller voir ailleurs en interrogeant sa mémoire. Ce désir intervient aussi plus fréquemment pour les actions boire et repos long car ces deux actions ne se font qu'en un nombre très restreint de lieux sur la parcelle. Dans ces cas il est nécessaire que le troupeau effectue un déplacement global et nécessite donc que l'animal voulant se déplacer soit déclaré leader pour être suivi par le reste du troupeau, sinon il devra se rabattre sur une action de proximité.

Cette relation de leadership et tout ce qui en découle n'est pas encore implémenté dans le prototype du simulateur sur lequel j'ai travaillé. Ainsi seul le grégarisme est présent dans cette version.

1.4 Etat de l'implémentation du simulateur : prototype 2003

Après avoir réalisé la modélisation du système étudié, modèles V, A et S, il fallait ensuite créer un programme informatique, appelé simulateur, qui permette que réaliser des simulations de grande envergure. Cette implémentation a été effectuée en C++ après être passé par une phase de modélisation utilisant la représentation UML. Actuellement, à l'exception de la mémoire des ruminants, l'ensemble du simulateur a été implémenté.

1.4.1 Le modèle V

Il fût entièrement implémenté, au centre INRA de Clermont-Ferrand-Crouël, et il comporte les trois fonctionnalités suivantes :

- mise à jour des cellules de la parcelle (croissance et sénescence)
- prise en compte de la défoliation des cellules par les animaux
- mise à disposition d'informations concernant l'état des cellules aux animaux

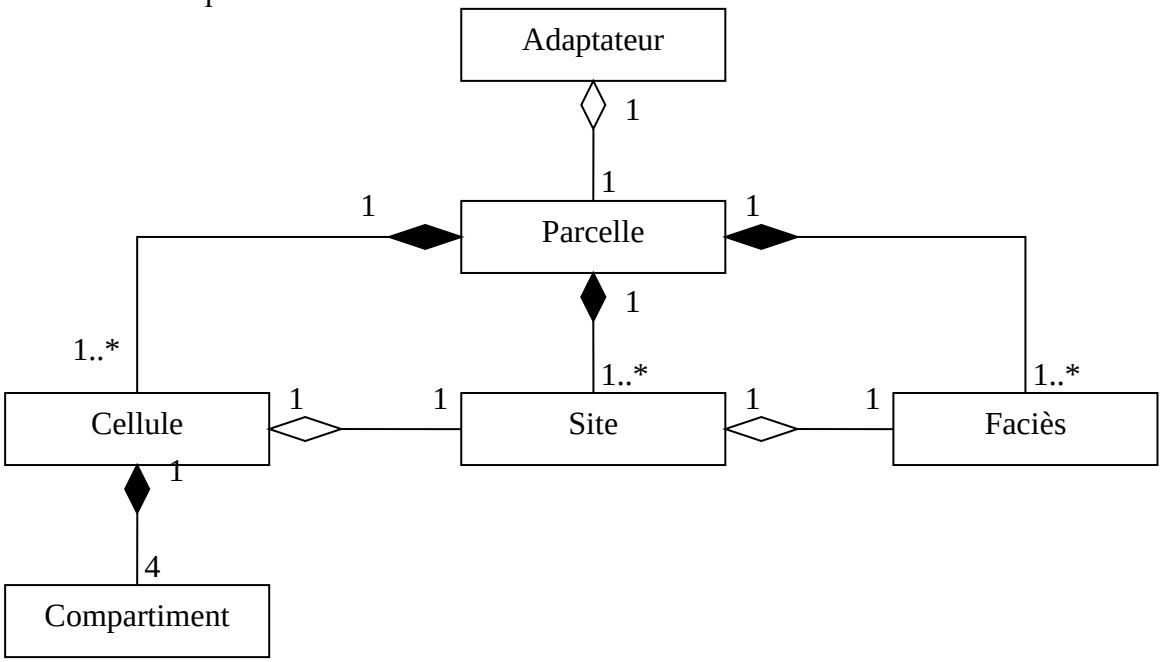

Figure 1-5 : Diagramme de classe du modèle Végétal

Dans ce diagramme (Figure 1-5) on retrouve les choix de structuration des diverses segmentations de la parcelle (voir 1.3.1.1). La parcelle est reliée à la classe Adaptateur qui assure la liaison avec le reste du simulateur (voir 1.4.4).

1.4.2 Le modèle A

Ce modèle a été implémenté suivant le diagramme de classe (Figure 1-6) suivant :

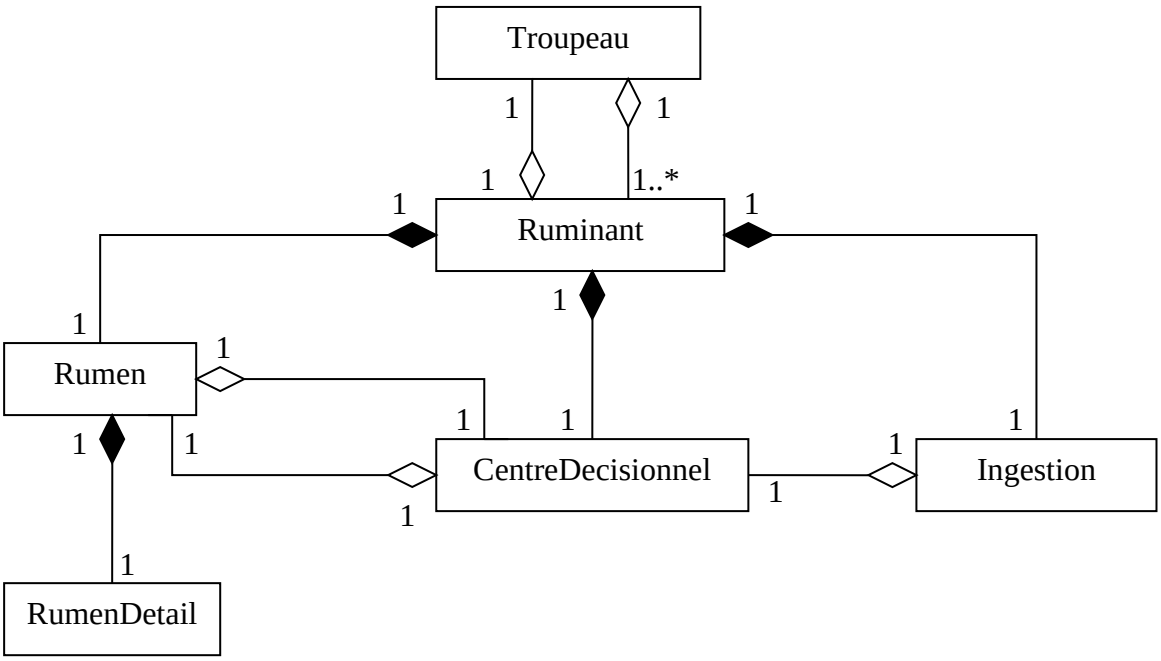

Figure 1-6 : Diagramme de classe du modèle Animal

1.4.3 Le modèle S

Ce modèle a été implémenté en partie (pas de mémoire spatiale) en commun avec le modèle animal. Ce modèle a surtout impliqué l'ajout de nouveau évènements comme les tests d'éloignement leader ou troupeau et test de rapprochement.

1.4.4 Liaisons entre les modèles

La liaison entre les modèles A et V (Figure 1-7) est réalisé par deux classes, la première « Adaptateur » sert d'interface entre les deux modèles, c'est-à-dire que c'est par là que vont passer toutes les informations qu'ils doivent s'échanger. La deuxième classe, « Information », a quant à elle le rôle de contenir toute les informations utiles d'une cellule de la parcelle, et c'est donc un tableau d'instances de cette classe qui sera fourni au modèle animal par le modèle végétal à demande d'information.

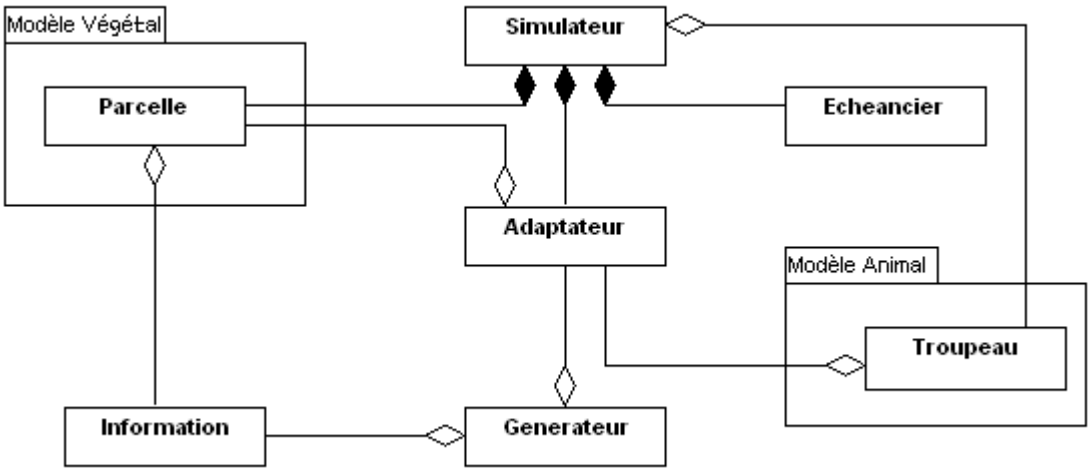

Figure 1-7 : Diagramme de classe de l'interconnexion entre modèles

1.4.5 L'échéancier

Dans la gestion du temps avec l'échéancier, une simulation par évènements discrets a été choisie, chaque évènement correspondant à une action d'un animal. L'unité de temps élémentaire de cet échéancier est la seconde.

Pourquoi avoir choisi cette approche plutôt qu'une horloge classique ? Le simulateur contient un ensemble d'entités (cellules et ruminants) qui peuvent commencer une action à n'importe quel moment (pour les ruminants) ce qui impliquerait de tester s'il y a un évènement à exécuter toutes les t secondes si une horloge avait été choisie. Alors qu'avec l'échéancier, lorsque aucun évènement n'est attendu dans un certain lapse de temps alors on le saute (ce qui nécessiterait un nombre considérable de tests avec un mode « horloge »).

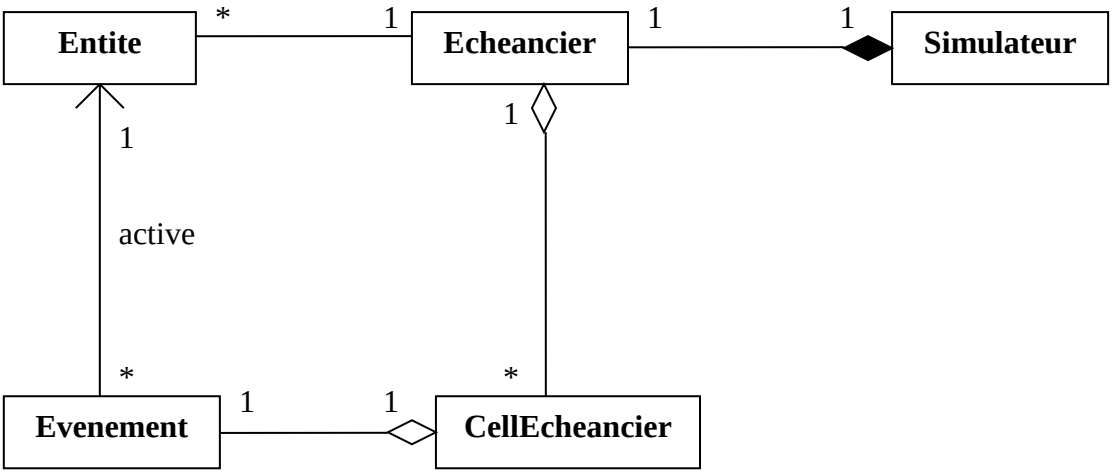

Figure 1-8 : Diagramme de classe de l'échéancier

L'échéancier correspond à une liste d'évènements, chacun étant associé à une date d'activation, c'est la classe CellEcheancier (ainsi chaque CellEcheancier est associé à un évènement). Enfin ce dernier active l'entité associé lorsqu'il est activé, et lors de son exécution l'entité peut modifier l'échéancier afin d'y introduire ses prochains évènements.

1.5 Parties validées

Le projet de développement du simulateur a démarré il y a quatre ans. Le projet a été conçu par phases de développement (segmentation). En effet, chacun des modèles ont été implémentés séparément, ce qui a permis des étapes de validation intermédiaires. Ainsi C. Pissavy [8] a réalisé une première validation du modèle V, en effectuant tout d'abord une paramétrisation donnant des valeurs standards à chacun des paramètres constitutifs du système (fichiers constantes, environnement et faciès). En faisant ensuite une estimation de la validité des sorties par comparaison des valeurs prédites de manière théorique et celles données par le simulateur. Et enfin elle a réalisé une analyse de sensibilité sur une grande partie des paramètres du modèle végétal. A l'issue de cette analyse, il a été conclut que le modèle végétal fonctionnait en autonomie et donnait des résultats cohérents.

Par la suite, une première étape de validation a été lancée sur le modèle animal (sans comportement social mais avec le modèle végétal) par M. Jouven [6]. Elle a obtenu un calibrage des principaux paramètres de l'animal et elle a fait ressortir certains défauts dont la vitesse d'ingestion qui variait trop brutalement entre un couvert épié et un couvert végétatif, sont apparus également des incohérences dans la gestion de la montaison dans le modèle végétal (absence de contrôle de la montaison par le pâturage) où il fallait ajouter un contrôle de cette fonction par les défoliations de l'animal.

1.6 Objectifs du stage

En l'état actuel du travail certains éléments n'ont toujours pas été implémentés, c'est le cas de la mémoire spatiale des ruminants dans le modèle S (c'est le travail d'un autre stagiaire en stage en même temps que moi [1]), et aussi le cas de l'échelle 3 de l'outil visionneuse. Dans le modèle A les fonctions permettant de choisir les activités de prise d'eau et de repos longs impliquant des déplacements longs n'ont pas été non plus encore implémentées. Mon travail de stage a pour objectif de vérifier et de commencer de valider une première version réunissant les modèles V, A et S (sans la mémoire spatiale et les déplacements longs), soit le fonctionnement d'un groupe d'animaux pâturant une petite parcelle (dans ce cas les fonctions mémoire et déplacement long sont moins fondamentales). Ce prototype développé par S. Portal sera appelé dans la suite prototype 2003.

J'ai tout d'abord commencé par apporter des améliorations au système de fichiers de sortie du simulateur en vue de simplifier leur utilisation avant de lancer une phase de vérification. En effet, depuis le stage de validation de M. Jouven [6] des modifications ont été apportées au simulateur, et elles n'ont pas été vérifiées. Ainsi il m'était nécessaire d'effectuer cette phase avant de lancer le travail de validation afin d'avoir un prototype totalement opérationnel sur lequel travailler. Une fois ce travail réalisé j'ai pu effectuer certaines modifications qui se sont imposées à l'issue du stage de M. Jouven.

2 Amélioration de la gestion des fichiers de sortie

Afin de pouvoir lancer la phase de vérification puis celle de validation du simulateur de manière efficace il a fallu au préalable effectuer un travail de remise en forme des sorties du simulateur. En effet, celles-ci possédaient de nombreux défauts comme le fait qu'elles ne soient pas uniformes dans leurs formats. Presque tous les fichiers de sortie contenaient des données sous forme de tableau, mais ils n'avaient pas tous les mêmes séparateurs (« , », « ; », « » ou tabulation). Cette absence d'homogénéité de structure était un obstacle pour créer facilement des macros ou autre outil de traitement automatique de ces sorties. Un autre problème était celui de la redondance de certaines données de sorties.

La gestion de la création des fichiers avant et après modification est représentée en annexe (?).

2.1 Les modifications apportées

2.1.1 Sur les sorties relatives à l'animal

J'ai d'abord changé les extensions des fichiers pour qu'ils aient tous l'extension *.csv* pour uniformiser le format des fichiers et faciliter leur utilisation (surtout sous Linux). J'ai également uniformisé en remplaçant tous les séparateurs des éléments des tableaux par des « ; ».

Tous les fichiers ont été renommés afin de raccourcir leurs noms en ne gardant que les informations utiles, en effet certains fichiers contenaient dans leur nom le préfixe *sortie* alors que le répertoire ne contient que des fichiers de sortie.

Voici les changements de nom que j'ai effectué :

- *sortieCentreDecisionnel.txt* devient *CentreDecisionnel.csv*
- *sortieIngestion.txt* devient *Ingestion.csv*
- *sortieRumenDetail.txt* devient *RumenDetail.csv*
- *sortieRumen.txt* et *calculRumen.csv* deviennent *Rumen.csv*
- *sortieSocial.txt* devient *Social.csv*
- *(nom_sim)(nom_methode).trj* a disparu
- *(nom_sim)(n°jour)(nom_methode).trj* devient *(n°jour)Percept(nom_methode)_trj.csv*
- *(nom_sim)(nom_methode).cel* devient *Percept(nom_methode)_cel.csv*
- *(nom_sim)(nom_methode).csv* devient *(n°jour)Percept(nom_methode).csv* et *Percept(nom_methode)Stat.csv*
- *AnimJour.csv* est apparu

Avec désormais *(nom_methode) = OFT* ou *Pref* au lieu du nom complet de la méthode qui était trop long et encombrait le répertoire rendant complexe le repérage d'un fichier spécifique.

Dans le cadre de l'analyse des fichiers de sortie, j'ai pu constater que les fichiers *sortieRumen.txt* et *calculRumen.csv* étaient les mêmes à l'exception de six colonnes supplémentaires présentes dans *calculRumen.csv*. Dans un premier temps j'ai corrigé une erreur dans le calcul de ces colonnes, en effet les valeurs étaient cumulées sur toute la durée de la simulation et le calcul ne se faisait pas à partir des bonnes valeurs. Ensuite j'ai supprimé l'écriture du fichier *calculRumen.csv* après avoir déplacé les six colonnes supplémentaires dans la classe RumenDetail, où *sortieRumen.txt* était écrit. Puis la classe CalculRumen qui était devenue inutile a été supprimée. Enfin les valeurs utilisées pour les durées des séquences d'ingestion et de rumination étaient fixées respectivement à 20.12 et 20 dans la fonction

Rumen::miseAJour() alors qu'elle sont gérées de façon variable dans le reste du modèle Animal, j'ai donc récupéré la valeur utilisée dans le reste du modèle dans cette fonction.

De plus j'ai uniformisé tous les fichiers contenant une date autour du format « Jour » puis « Secondes » où les secondes sont celles depuis le début du jour courant, sauf pour les fichiers journaliers. En effet, avant ce changement il arrivait que les secondes soient celles depuis le début de la simulation rendant moins rapide le repérage dans le temps puisque que l'on est alors obligé de calculer le jour courant.

Les autres modifications que j'ai apportées aux fichiers de sortie sont spécifiques à chaque fichier :

- ✓ *Percept(nom_methode)_cel.csv* : Dans ce fichier la date n'était écrite que de temps en temps ce que j'ai rectifié en l'écrivant pour toutes les lignes, de plus j'ai renommé les colonnes de manières compréhensible car avant leurs libellés étaient « param i » pour i de 1 à 5.
- ✓ *Social.csv* : J'ai modifié ce fichier pour qu'il commence à ressembler à un tableau, pour cela je l'ai écrit sous forme d'un tableau de deux colonnes avec des lignes vides pour séparer chaque évènement.
- ✓ *Echeancier.csv* : Ce fichier était écrit sous forme de ligne de texte inexploitable et je l'ai donc transformé en un tableau dont les valeurs correspondent aux nombres d'occurrence sur la journée alors qu'avant elles étaient cumulées et donc difficilement compréhensible au premier abord.
- ✓ *Ingestion.csv* : Ce fichier qui ne contenait avant que des informations sur la première défoliation de chaque séquence d'ingestion, contient désormais une ligne par séquence d'ingestion avec des moyennes et en plus la durée de cette séquence. J'ai aussi modifié le compteur qui permettait de connaître le nombre de défoliations par séquence car il contenait des valeurs cumulées sur toute la simulation puisque l'attribut utilisé dans le code était en « static ». Enfin j'ai rajouté pour chaque site de la parcelle les colonnes suivantes :
    - NbChoix
    - Temps = somme(TempsCell)
    - QI = somme(QICell)
    - NDFseq
    - DNDFseq

Ensuite j'ai créé un nouveau fichier résumant l'activité de chaque animal par journée, *AnimJour.csv*. Ce fichier est donc constitué d'une ligne par journée de simulation animale avec les informations suivantes :

- ✓ Jour
- ✓ Ding_cumulees (durée d'ingestion cumulée sur la journée)
- ✓ Qing_cumulees (les quantités ingérées par l'animal sur la journée)
- ✓ Drum_cumulees (durée de rumination cumulée sur la journée)
- ✓ moyenne de NDFseq
- ✓ moyenne de DNDFseq
- ✓ moyenne de QIseq (la moyenne des durées d'ingestion sur une séquence sur une journée)
- ✓ moyenne de LP
- ✓ moyenne de SP

<u>NB</u> : les moyennes de *NDFseq*, *DNDFseq* et *Qiseq* sont calculées uniquement pour les valeurs non nulles des variables concernées.

Enfin j'ai fait réapparaître un fichier qui avait été désactivé, *(nom_sim)(nom_methode) .csv*, ce fichier contenait des statistiques sur les choix de cellules du ruminant par séquence d'ingestion. Elles sont calculées dans la classe CalculStats et il avait été placé en commentaire lors de son appel dans la fonction *main()*. Je l'ai ensuite séparé en deux fichiers *(n°jour)Percept(nom_methode).csv* et *Percept(nom_methode)Stat.csv*. Le deuxième est issu des données fournies dans le fichier d'origine sous forme de texte, donnant des informations sur le nombre de choix suivant les directions, les rangs et les sites, il contient donc une ligne par journée de simulation. Les autres fichiers (un par jour), contiennent les informations suivantes : site, abscisse, ordonnée, choix cellule, qualité cellule, probabilité globale, probabilité sélection, probabilité qualité et probabilité direction/rang. De plus j'ai apporté certaines modifications pour permettre une utilisation correcte de la classe CalculStats avec plusieurs animaux.

Mais il s'avéra après le lancement de nombreuses simulations que ce traitement effectué après la simulation mettait beaucoup trop de temps et serait plus performant en l'incorporant dans le simulateur car on refait après la simulation la lecture des sites dans le fichier *carte.def* et on relit certains fichiers qui sont écrits pendant la simulation. Ainsi il était évident que l'on gagnerait en performance en plaçant l'écriture des deux fichiers là où sont écrits les fichiers dont ils utilisent les valeurs. Désormais le fichier *(n°jour)Percept(nom_methode).csv* est crée dans Generateur<T>::choixCellule() et le fichier *Percept(nom_methode)Stat.csv* dans Ruminant::ecritStat().

2.1.2 Sur les sorties végétales

Parmi les fichiers de sortie du modèle Végétal j'ai ajoutés deux fichiers *Maj_Init.csv* qui est un fichier identique à *Maj_(n°jour).csv* mais qui correspond à la configuration initiale de la parcelle et *StatsVege.csv* qui est un fichier de statistique sur l'état de la parcelle pendant la simulation. Ce dernier fichier possède une ligne par jour et par site avec une colonne pour la moyenne et une pour l'écart-type de toutes les variables présentent dans le fichier *Maj_(n°jour).csv*. Pour les calculs j'ai eu besoin d'utiliser des fonctions statistiques particulières pour la moyenne et l'écart-type afin de limiter le temps de calcul, j'ai pris la formule suivante pour l'écart-type :

$$\sigma = \sqrt{\left|\frac{1}{n-1}\left(\overline{X^2} - \frac{\overline{X}^2}{n}\right)\right|}, \text{ pour } n > 1$$

Le deuxième traitement de sortie du modèle végétal porte sur l'autre catégorie de fichiers en sortie qui correspondent aux informations qui sont transmises entre les deux modèles. Côté modèle végétal, j'ai séparé la première colonne du fichier *DefolTemp.csv* en trois colonnes : Ligne;Colonne;Wvv et supprimé les « - » du fichier *Defol.csv*. Côté modèle Animal (l'adaptateur), j'ai ajouté la fin de la ligne de titre du fichier *Choix.csv* et les titres des variables : Ligne;Colonne de *Defol.csv*. De plus ces deux fichiers étaient écrits ligne par ligne dans des fichiers temporaires dans le modèle végétal pour être ensuite recopiés dans le fichier final dans le modèle animal, cela ralentissait l'exécution du simulateur sans nécessité apparente, en effet l'écriture dans un fichier est une action très chère en temps d'exécution et doit par conséquent être évitée dès que cela est possible. J'ai donc supprimé l'écriture des fichiers temporaires en écrivant directement les fichier définitifs et par la même occasion j'ai fait en sorte que ces deux fichiers ne soient pas créés lorsqu'il n'y a aucun animal puisqu'ils décrivent les informations transmises entre les deux modèles.

Enfin comme pour le modèle Animal tous les séparateurs des fichiers de sortie du modèle végétal on été remplacés par des « ; ».

(Voir Annexe 7 pour un tableau résumant les fonctions et les noms de tous les fichiers de sortie)

2.2 Travail restant à faire

Le travail à effectuer sur les fichiers n'est pas terminé car un certain nombre de défauts sont encore à rectifier :

- Il faut finir de modifier le fichier *Social.csv* pour le mettre sous la forme de un à trois tableaux en colonne, pour cela on peut s'inspirer du rapport de projet de E. Begou et C. Descours [27].
- Il serait bon d'essayer de réunir les fichiers *(n°jour)Percept(nom_methode)_trj.csv* et *(n°jour)Percept(nom_methode).csv* qui fournissent tous les deux des informations par journée et par cellule ingérée. Mais attention, le premier de ces deux fichiers sert à la visionneuse et il peut ne pas être très intéressant à changer puisque cela risque d'empêcher l'utilisation de ces fichiers pour la visionneuse, ou alors il faudra modifier cette dernière pour prendre en compte ces changements.
- Le fichier *Choix.csv* possède encore des lignes avec des « - » qui peuvent peut-être disparaître sans que cela nuise à la lisibilité du fichier.
- Lors des tests que j'ai effectués j'ai pu constater que pour une simulation certains fichiers de sortie sont systématiquement utilisés alors que d'autres ne sont consultés qu'occasionnellement. Il serait donc intéressant de pouvoir sélectionner les fichiers de sortie désirés dans l'interface pour non seulement gagner de la place sur le disque dur mais aussi gagner du temps au niveau de l'exécution des simulations.
- Enfin, il serait bon de centraliser toutes les écritures des fichiers de sortie dans une seule et même classe pour augmenter la lisibilité du code et aussi faciliter une éventuelle nouvelle modification des sorties.

Il existe en parallèle du simulateur et de son interface, un petit utilitaire permettant de visualiser les déplacements de l'animal et l'état de la parcelle, il s'agit de la visionneuse. Or celui-ci utilise certains fichiers de sortie du simulateur qui ont été légèrement, voire beaucoup modifiés ici. Il sera donc sûrement nécessaire de modifier la lecture de ces fichiers dans la visionneuse. Ces modifications ont été en partie apportées par R. Martin [1] qui a incorporé la visualisation de la mémoire pour l'échelle 2 et dû donc prendre en compte mes modifications des fichiers de sortie du simulation.

3 Vérification du « prototype 2003 »

Ainsi grâce à cette phase d'amélioration des sorties du simulateur, j'ai non seulement amélioré l'efficacité de futurs traitements sur ces sorties et aussi je me suis familiarisé avec le code du simulateur ce qui m'a permis une plus grande efficacité dans mes recherches dans le code. Sur ce dernier point j'ai pu profiter du travail de R. Martin [1], qui a réalisé une documentation dynamique du code source à l'aide de l'outil doxygen (voir Annexe 2). Celle-ci permet de suivre les appels de fonctions entre les différentes classes du programme, elle contient également le rôle de ces fonctions, la signification des variables, etc.

Ainsi une fois ce premier travail réalisé, j'ai pu m'attaquer à la vérification du « prototype 2003 » dont le code est issu du stage de S. Portal [5]. En effet, lors de son stage elle a apporté un certain nombre de modifications qui n'ont pas été vérifiées. Puisque mon travail a pour objectif la validation de ce prototype, je devais au préalable vérifier le code afin de garantir son bon fonctionnement. J'ai ensuite comparé les résultats obtenus avec ceux obtenus par M. Jouven avec les mêmes jeux de paramètres, mais en utilisant un prototype antérieur du simulateur.

Lors de son stage S. Portal a travaillé sur le modèle Social et Spatial du simulateur qui n'était pas présent dans le prototype sur lequel a travaillé M. Jouven. Ainsi elle a modifié tout le système de gestion des évènements en utilisant l'échéancier (1.4.5), elle a aussi introduit un automate décisionnel plus complet (voir 1.3.2.1), etc.

Cette démarche permet de vérifier d'une part qu'aucune autre imperfection ne soit apparue et que d'autre part les imperfections mises en évidences par M. Jouven aient bien été rectifiées.

Afin de réaliser cette vérification efficacement, j'ai commencé par m'intéresser aux modifications apportées par S. Portal de manière indépendante. Ainsi j'ai réalisé des diagrammes d'activité afin de comparer le fonctionnement de ce qui était implémenté avec ce qui était prévu par le modèle conceptuel. Ensuite j'ai réalisé des séries de simulations en testant un grand nombre de combinaisons de paramètres qui m'ont permis de repérer des imperfections comme un manque de tests de sécurité sur les paramètres d'entrée. A partir de là, des améliorations ou corrections se sont imposées que nous allons voir dans les paragraphes suivants (3 et 4). Après cela, j'ai lancé une série de simulations dans des conditions standard (c'est-à-dire avec les valeurs des paramètres issues des calibrations réalisées lors des stages précédents) qui m'ont permis de vérifier la cohérence des valeurs obtenues en vérifiant si elles sont dans le domaine possible et biologiquement plausibles (par exemple un mouton ne mange pas plus de 3000 g de MS par jour). C'est-à-dire que l'on ne regarde ici que l'absence d'erreurs flagrantes alors que dans la phase suivante on s'intéressera à une cohérence plus poussée en effectuant des comparaisons avec la réalité que l'on veut modéliser.

3.1 Vérification de l'automate décisionnel

Dans un premier temps je mes suis intéressé à l'automate décisionnel que S. Portal a implémenté. Dans cette version (voir 1.3.2.1) des valeurs étaient calculées pour chaque action et étaient ramenées entre 0 et 1 pour servir de probabilités mais cette méthode ne garantissait pas une assez grande priorité à l'action manger lorsque celle-ci devenait indispensable. Pour pallier à ce problème, j'ai modifié l'automate décisionnel. Les figures Figure 3-1 et Figure 3-2 décrit la façon dont est actuellement déterminée l'action à effectuer.

C'est pour cela que j'ai modifié cet automate pour prendre en compte ce constat, désormais voici comment on détermine l'action à effectuer :

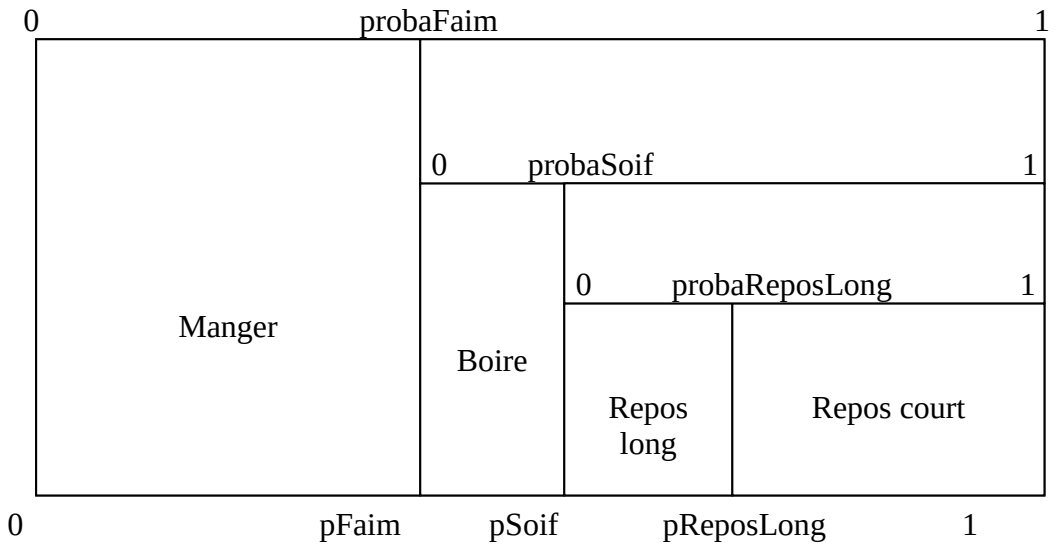

Figure 3-1 : Schéma du nouvel automate décisionnel

Avec :

$$pFaim = probaFaim$$

$$pSoif = probaSoif \times (1 - probaFaim) + probaFaim$$

$$preposLong = (probareposLong(1 - probaSoif) + probaSoif)(1 - probaFaim) + probaFaim$$

Cette fois-ci les valeurs de probaFaim, probaSoif et probaReposLong sont directement des probabilités. Cela simplifiera également les calculs des probabilités de boire ou de faire un repos long qui devrait être réalisé lors d'un travail ultérieur (ces deux actions ne seront pas incluses dans le prototype que j'utilise lors de mon stage). Ce nouvel automate est en faite un mélange entre les deux automates déjà envisagés ([5] page 15) car on redonne un ordre de priorité entre les actions : d'abord manger puis boire, puis repos long et enfin le repos court tout en gardant une partie aléatoire dans la décision de l'animal.

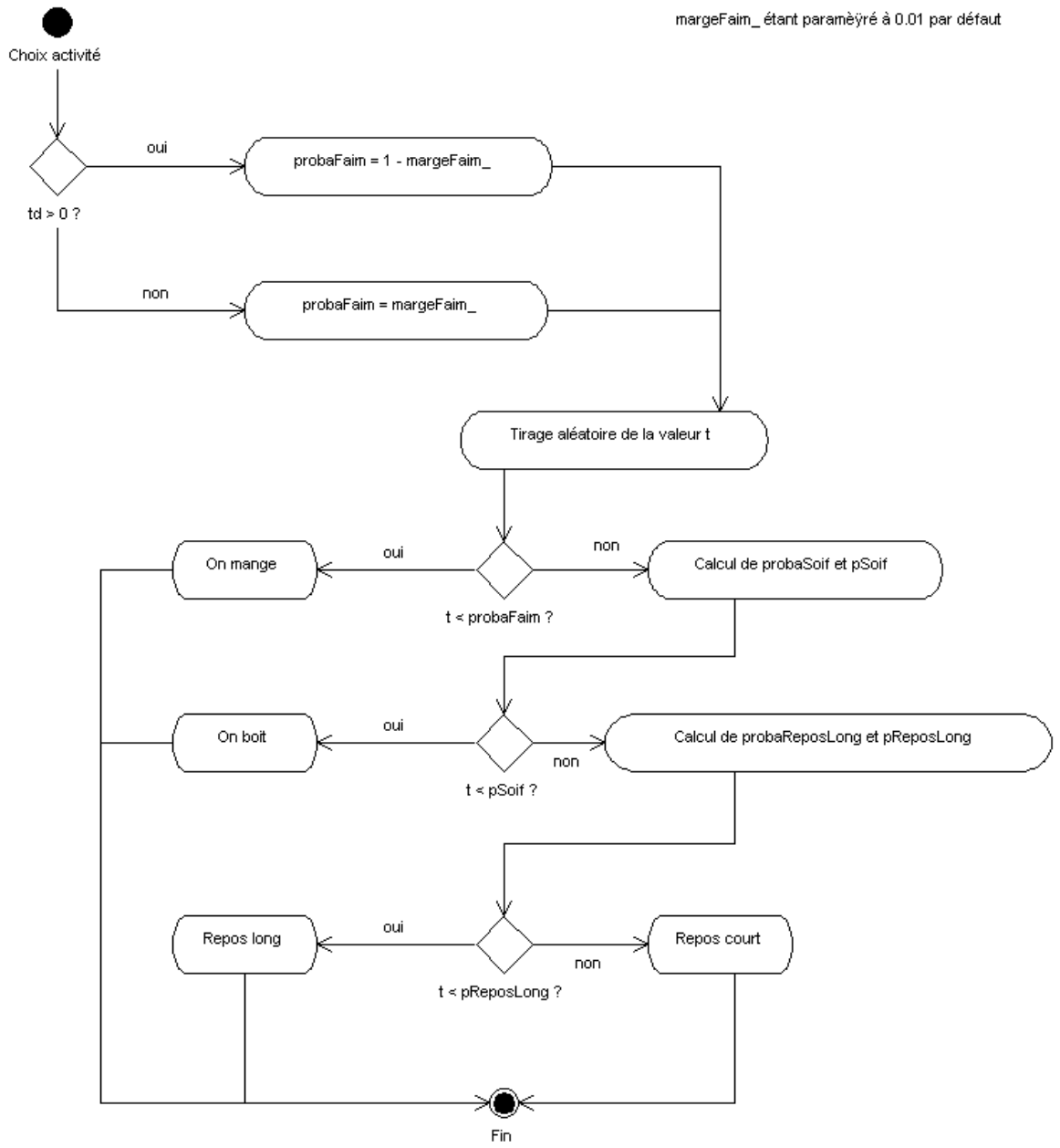

Figure 3-2 : Diagramme d'activité pour le nouvel automate décisionnel

D'autre part le calcul de la probabilité de manger a été modifié pour se mettre en accord avec la modélisation du choix de l'activité « manger » et corriger une mauvaise gestion de la nutrition de l'animal. En effet, l'animal mangeait très irrégulièrement d'une journée à l'autre, pendant une journée l'animal mangeait une quantité x et le lendemain il en mangeait 2x provoquant des oscillations sur toute la durée de la simulation. Après avoir analysé plus en détail ce phénomène je me suis rendu compte que l'animal ne mangeait pas assez, x était bien inférieure à la quantité nécessaire pour satisfaire les besoins nutritionnels. Ainsi le jour suivant la fonction de motivation à ingérer augmente à cause du manque de la veille et l'animal mange donc beaucoup plus et ainsi de suite pour les jours suivants. A partir de cette analyse, il devint évident que l'erreur venait de l'automate décisionnel et plus précisément de la probabilité de manger. Celle-ci était calculée dans le code autour du test Fmi-Fsat>=0 alors

que dans la modélisation (Sauvant et al. [24], Baumont et al. [25]) le test était Fmi-Fsat >= -0.5. Mais en plus de cette correction, on a changé le test en td > 0, avec :

$$td = (Fmi - Fsat)(0.8 + tiragteAléatoire \times 0.4) + poidsManger$$

Où poidsManger est un paramètre qui vaut par défaut 0.5 (valeur standard). Le tirage aléatoire s'effectue entre 0 et 1.

Lors de ce calcul de probabilité on a utilisé un attribut margeFaim_ (qui est paramétré par défaut à 0.01) de manière à obtenir une probabilité de manger de 0.99 si l'animal a faim (lorsque le test précédent renvoie 1) et 0.01 sinon, ce qui assure que l'animal va manger lorsqu'il a faim et évitera ainsi les retards d'alimentation qui se produisaient.

Ensuite je me suis intéressé au traitement de l'action ruminer. Dans le modèle conceptuel, l'animal peut ruminer lorsqu'il est en repos court ou en repos long selon une probabilité de rumination. Or après avoir analysé comment était géré la rumination dans le « prototype 2003 », je me suis rendu compte que non seulement la probabilité de rumination n'était pas utilisée mais aussi que l'animal ruminait quand il devait boire (même si cette action n'était pas implémentée). (Rq : la version sur laquelle j'ai travaillé ne contenait que les actions manger, repos court et rumination, à l'avenir il faudra aussi intégrer la possibilité que l'animal rumine pendant les phases de repos long). J'ai donc rectifié cela en gérant la rumination après le choix de l'activité principale (manger ou repos court dans mon cas), c'est-à-dire au moment où on appelle les fonctions qui gèrent les évènements (fonction Ruminant::choixActivite()). Contrairement aux autres actions, la rumination n'est pas gérée comme un évènement, elle est associée à l'action repos court, elles ont donc la même durée. Elle est donc réglée par un seul attribut de la classe Ruminant, rum_, qui vaut 1 lorsque la rumination doit être réalisée (détermine le traitement à effectuer par la fonction Rumen::miseAJour()). Ainsi j'ai traité la rumination de la manière suivante :

```
Si on a choisi l'action manger alors
 On appelle mange() (gère l'évènement manger)
Sinon (on a choisi le repos court)
 On calcule la probabilité de ruminer
 On effectue un tirage aléatoire
 Si le tirage est plus petit que la probabilité de ruminer alors
      rum_ = 1 (l'animal rumine)
 FinSi
 On appelle reposCourt() (gère l'évènement repos court)
FinSi
```

3.2 Autres corrections apportées

Ensuite j'ai repéré et corrigé un certain nombre d'erreurs dont voila la liste :

- Une erreur d'unité dans le calcul de la quantité de biomasse que l'animal va défolier sur une cellule, ce calcul se faisant à partir de la formule suivante :

  $$QiCell = \frac{TempsCell \times ViPermise}{Fsat}$$

  Où la valeur de *TempsCell* (temps que l'animal va passer sur la cellule) était fournie en minutes alors que *ViPermise* (la vitesse maximale qui est autorisée par la cellule) était en quantité par secondes. Cette erreur donnait des valeurs aberrantes des quantités ingérées, et par conséquent des durées d'ingestion énormes sur plusieurs journées. J'ai donc remis tempsCell en secondes.

- Par la suite lors de mes premiers essais avec plusieurs animaux j'ai constaté qu'il n'y avait aucun test de sécurité sur les paramètres de grégarité du troupeau, la fonction de distance et son paramètre associé. En effet, pour la fonction « barycentre » on pouvait mettre un nombre d'animaux pour le calcul supérieur à leur nombre dans le troupeau ce qui faisait planter le simulateur, j'ai donc incorporé ces tests et les ai accompagnés d'un message donnant le domaine de validité de ces paramètres.

J'ai également corrigé plusieurs petits défauts tels que la surcharge inutile du code avec des variables obsolètes ou des décalages d'indices dans des tableaux. Il y avait aussi un guide de style à respecter dans la mesure du possible et aussi des commentaires à compléter à la manière dont R. Martin les a réalisé pour l'outil doxygen (voir Annexe 2 et 3).

Enfin après toutes ces modifications il reste certains éléments à éventuellement modifier car ils pourraient améliorer le simulateur dans son utilisation :

- Vérifier les valeurs d'initialisation des variables du rumen : *lp, sp, cclp, dndflp, ccsp, dndfsp* et *vfa.*
- *Preference1* et *Preference2* sont identiques, si cela doit toujours être le cas, il n'est pas utile de les garder toutes les deux.
- En ce qui concerne le fichier *param.csv*, il y a une redondance des données avec *carte.def* et *facies.csv* qui peut occasionner des erreurs. Il serait donc judicieux par exemple de créer à chaque fois le fichier *carte.def* (par l'interface si on utilise les types de carte prédéfinis) afin de lire les dimensions de la carte dans ce fichier plutôt que dans *param.csv*. D'autre part il faut bien préciser dans l'interface au moment de la saisie de *param.csv* que le nombre de faciès demandé correspond au nombre de lignes du fichier *facies.csv* et non au nombre de faciès utilisés.
- Il serait également judicieux de réduire la taille du fichier *carte.def* car il contient trop de commentaires, ces derniers pourraient être mis dans un fichier *lisez-moi.txt* dans le même répertoire
- Lors de certaines simulations, il s'est avéré que les quantités de biomasses à ingérer par compartiment envoyées par le modèle Animal au modèle Végétal pouvaient valoir « nan » (Not A Number) ce qui se produit lorsqu'on effectue un calcul avec une variable valant « inf » (Infinity). En remontant les appels de fonctions j'ai constaté que cela venait du calcul de la quantité à ingérée sur la cellule où la fonction de satiété valait « inf ». Cela était du à une absence de nutrition de l'animal pendant plusieurs jours ce qui provoque des variations trop importante. Je n'ai pas corrigé ce disfonctionnement puisque je ne l'ai découvert qu'à la fin de mon stage.

4 Améliorations apportées aux modèles biologiques

Après mon travail de vérification du « prototype 2003 », j'ai apporté des modifications dans le code du simulateur. Ces modifications étaient rendues nécessaires suite à des erreurs conceptuelles apparues après la validation réalisée par M. Jouven [6]. En effet, lors de son travail elle a mis en évidence certains défauts au niveau de l'évolution de la vitesse d'ingestion et aussi sur les fonctions sénescence et montaison du modèle végétal. Afin de corriger ces imperfections il me fallait être sûr qu'elles existaient toujours dans la version sur laquelle je travaillais. C'est pour cela que dans un premier temps j'ai effectué un certain nombre de comparaisons entre les résultats que j'obtenais avec le « prototype 2003 » vérifié et ceux obtenus lors de la validation de M. Jouven. Ces comparaisons m'ont permis de m'assurer que les améliorations à apportées que M. Jouven avait soulevées étaient toujours nécessaire. Elles ont aussi soulevées des problèmes au niveau des paramètres d'entrées pour le modèle végétal qui n'étaient plus correctement calibrés du fait des nombreuses modifications apportées au simulateur depuis leurs premiers calibrage. J'ai de nouveau réalisé ce travail de calibrage.

4.1 Description des améliorations apportées

4.1.1 La vitesse d'ingestion

Elle était gérée de façon à utiliser deux fonctions différentes suivant la proportion de biomasse se trouvant dans le compartiment VV par rapport aux autres compartiments. Mais le passage entre les deux équations se faisait de façon brutale, c'est pour résoudre ce défaut qu'il a été introduit une continuité pour la transition entre ces deux fonctions. Désormais le calcul de la vitesse d'ingestion se fait de la façon suivante :

<u>si</u> $Brv + Brs \leq 0.10 \times Bm$ <u>alors</u>

$$vi = \frac{1000 \times mlv}{(24.9 + 153 \times mlv)} + 0.4596 \times \left(poids^{0.71} - 69^{0.71}\right)$$

<u>sinon si</u> $Brv + Brs \geq 0.25 \times Bm$ <u>alors</u>

$$vi = \frac{1000 \times mlv}{(46.8 + 202 \times mlv)} + 0.4596 \times \left(poids^{0.71} - 69^{0.71}\right)$$

<u>sinon</u>

$$vi = 1000 \times mlv \times \left(\frac{1}{24.9 + 153 \times mlv} - \frac{1}{0.15} \times \left(\frac{1}{24.9 + 153 \times mlv} - \frac{1}{46.8 + 202 \times mlv} \right) \times \left(\frac{Brv + Brs}{Bm} - 0.1 \right) \right) + 0.4596 \times \left(poids^{0.71} - 69^{0.71}\right)$$

Avec :

$mlv = \gamma_{vv} \times Bvv$

$Bm = Bvv + Brv + Bvs + Brs$

On obtient ainsi une transition progressive entre les deux courbes de la vitesse d'ingestion pour un couvert végétatif (vi1) et un couvert reproducteur (vi2) (figure Figure 4-1).

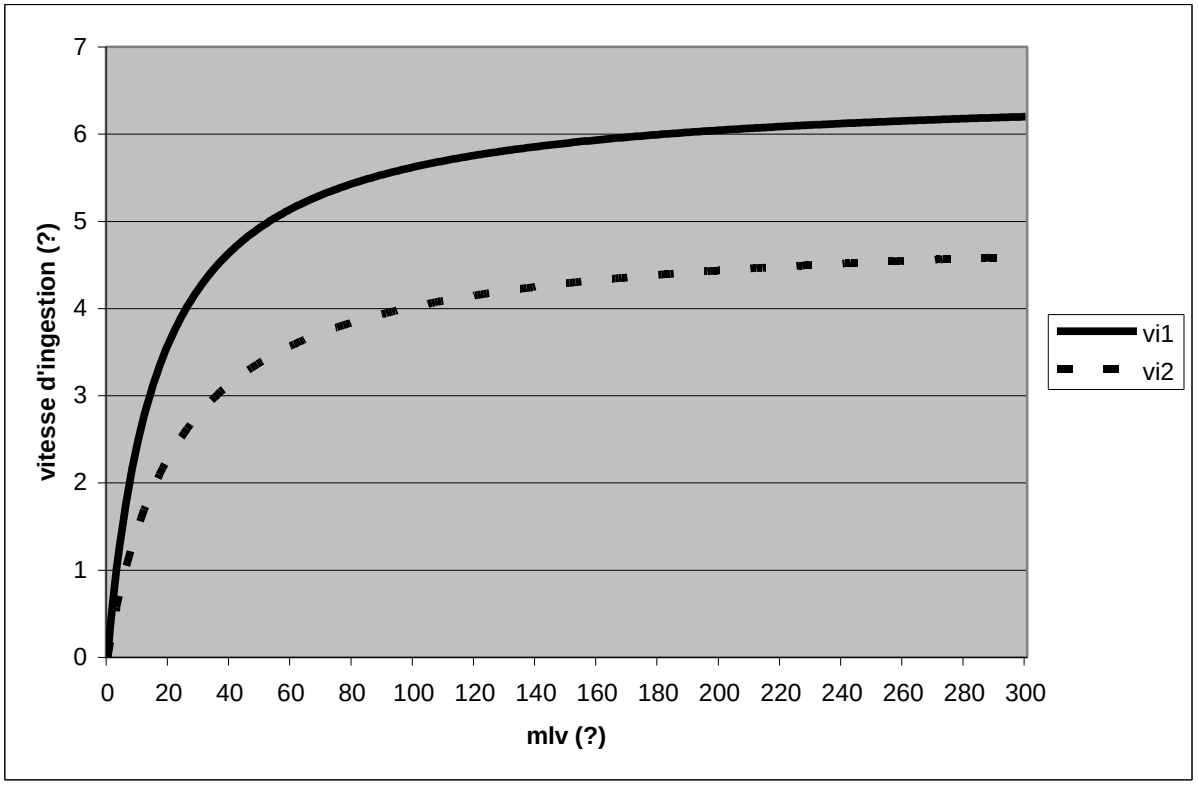

Figure 4-1 : Les deux courbes extrêmes de la vitesse d'ingestion en fonction de la masse de limbes verts

Où vi1 correspond au cas où $Brv + Brs \leq 0.10 \times Bm$ et vi2 au cas où $Brv + Brs \geq 0.25 \times Bm$

Dans le même temps, on a modifié le calcul de *mlv* (masse de limbes verts) afin de prendre en compte le γ_{vv} (proportion de limbes verts dans le compartiment VV) qui est calculé dans le modèle Végétal à la place du coefficient 0.5 qui était utilisé auparavant.

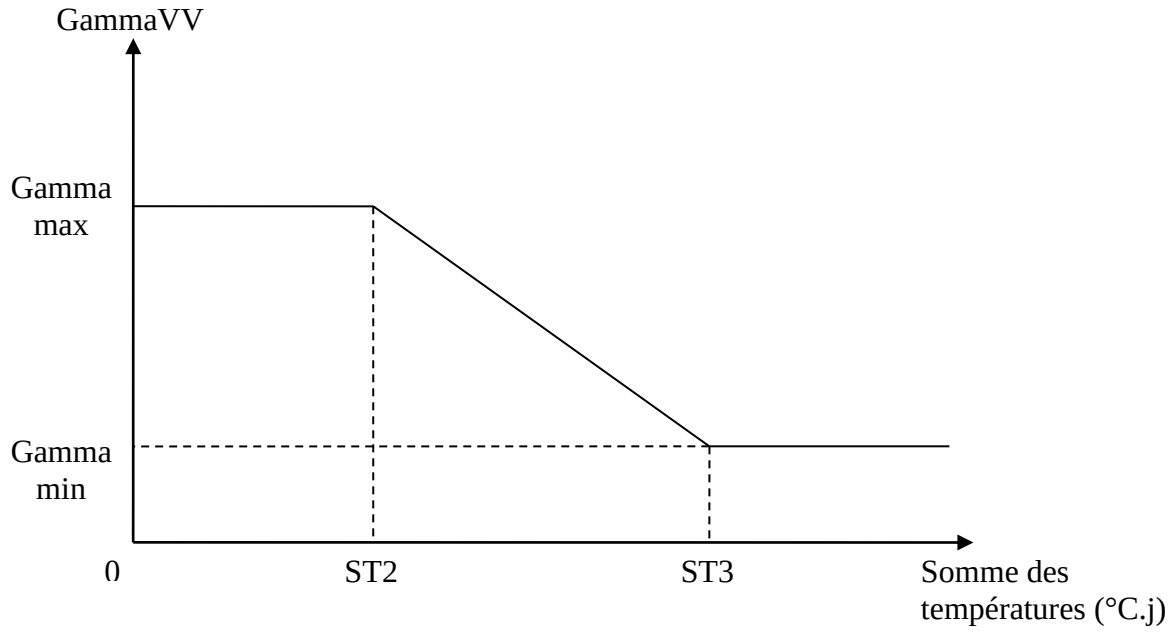

Figure 4-2 : Evolution de GammaVV en fonction de la somme des températures

ST2 et ST3 sont les bornes d'activation et de désactivation de la fonction montaison qui gère le passage du stade végétatif au stade reproducteur. Cette modification de la physiologie des plantes modifie grandement leur structure et donc la proportion de limbes verts dans la biomasse totale.

Les valeurs de ST2, ST3, Gamma min et Gamma max sont fournies en entrée du simulateur dans le fichier *facies.csv*.

Ces deux modifications interviennent dans le calcul de la vitesse d'ingestion (fonction Ingestion::calculTempsCellule()) et dans le calcul de la qualité perçue avec la fonction de perception Optimal Foraging Theory.

4.1.2 Les fonctions montaison et sénescence

Le stage de M. Jouven [6] avait fait apparaître deux problèmes majeurs : le manque de contrôle de l'épiaison sur les cellules défoliées, et le faible flux de sénescence généré par la végétation vieillissante. L'analyse conceptuelle réalisée par P. Carrère a été implémentée et incorporée dans le modèle végétal par Martin et Pean [9] . J'ai intégralement repris ces modifications dans la version que j'utilisais du simulateur.

4.2 Calibration

De nombreuses modifications qui ont été apportées depuis la version sur laquelle a travaillé M. Jouven [6] pour son stage de validation, les miennes, celles de S. Portal [5] et celles de R. Martin et B. Pean [9]. Ainsi certains calibrages qu'elle avait effectués pour l'animal et d'autres qui avaient été faits pour la végétation devaient être revus pour s'adapter aux changements du simulateur. Ainsi dans ce chapitre je présente le travail de calibration que j'ai réalisé sur le modèle végétal avec l'aide de P. Carrère. Ensuite je parle des valeurs standard qui seront utilisées pour la suite du travail pour le modèle animal et le modèle social.

4.2.1 Pour le modèle végétal

Le calibrage concernait ici les deux fichiers d'entrée standards suivants : environnement.csv et facies.csv, et le code des quatre compartiments des cellules.

Dans un premier temps, le fichier environnement.csv comportait des valeurs bien trop particulières pour des données « standard » car il contenait les valeurs d'une seule année. Celles-ci ont été remplacées par P. Carrère par les moyennes des températures observées sur la commune de Saint Genès Champanelle entre 1986 et 2002 (et de même pour le rayonnement solaire) (voir Annexe 4, la somme des température est exprimée en degrés.jour (°C.j)). Ce qui a lissé les données utilisées sur l'année afin de pouvoir calibrer ensuite sur des conditions standard. Cela a permis de déclencher la montaison à une période correcte, en effet avec les anciennes valeurs elle se déclarait beaucoup trop tard. De plus au cours de l'année précédemment utilisée, année froide, il y avait des gelées en Mai ce qui stoppait la croissance et limitait les quantités de biomasse disponible pour l'animal par rapport à des conditions standard. Ces conditions environnementales standard permettent en outre de s'abstenir de l'influence de la variabilité environnementale sur la parcelle pour ne se consacrer qu'à celle de l'activité des animaux, ce qui constitue l'objectif du travail.

Dans un deuxième temps l'équipe FGEP de l'INRA de Crouël a élaboré un nouveau fichier *facies.csv* standard qui m'a été fournit au début du stage avant de commencer son calibrage à la vue des dernières modifications du simulateur. Ce fichier contient désormais quatre faciès « type » :

1) Basé sur les données de F. Garcia (Dactyle) [26]
2) Basé sur l'expérimentation excédent (F. Louault [29]) pour une bonne intensité d'exploitation
3) Basé sur l'expérimentation excédent (F. Louault) pour une intensité d'exploitation plus faible (sans fauche)
4) Basé sur l'expérimentation excédent (F. Louault) pour une intensité d'exploitation très faible (un pâturage unique) D'autre part, toutes les valeurs correspondent à une situation de la parcelle au 1er Janvier.

Enfin nous avons pu commencer le calibrage des valeurs du fichier *facies.csv*. Après quelques tests nous avons mis en évidence une trop faible production de biomasse, on a donc cherché à rectifier ce problème en augmentant les biomasses des compartiments VV et RV. La première solution que nous avons envisagée était la plus évidente, travailler sur les entrées de croissance, SLA et INcell. Mais ces deux variables avaient déjà des valeurs très élevées et les augmenter encore signifiait un éloignement trop important par rapport aux données expérimentales et donc une perte au niveau de la cohérence par rapport monde réel. Cette solution a donc été écartée, on a même diminué le SLA pour se rapprocher des données expérimentales en faisant la moyenne des valeurs obtenues sur des couverts de dactyle cultivés en collection ce qui nous a donné un SLA de 0.026 $m^2.g^{-1}MS$ ([28]).

Ensuite on a envisagé de diminuer le flux de biomasse qui part vers le compartiment sénescent VS, pour cela on a réduit la valeur du paramètre Kvs en le passant à 0.001 au lieu de 0.002.

Ensuite on a accéléré la montaison en faisant passer la valeur du paramètre aIN de 0.20 à 0.25 et en modifiant la fonction de montaison dans le code du simulateur pour que f(IN) = 1 pour IN = 0.8 au lieu de 1.

$$f(IN) = \frac{(1 - aIN)}{(0.8 - IN0)} \times (IN - IN0) + aIN \text{, pour } IN0<IN<0.8$$

$$f(IN) = 1 \text{, pour IN>0.8}$$
$$f(IN) = aIN \text{, pour IN<IN0}$$

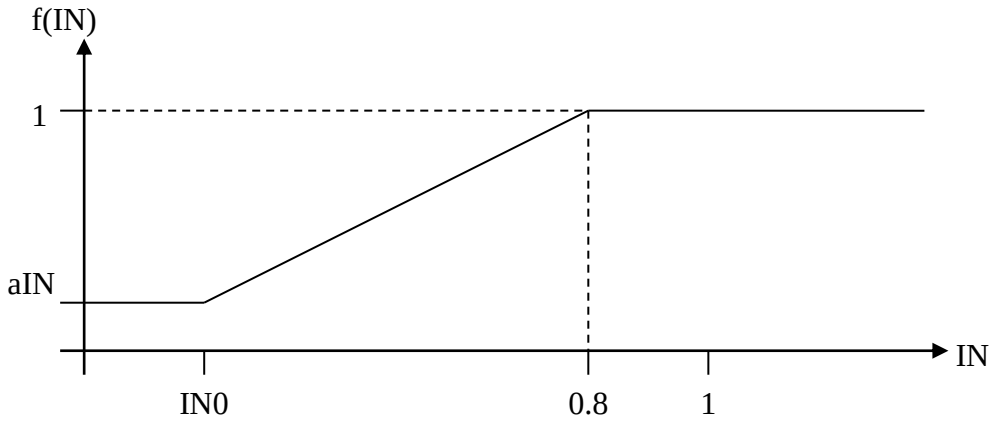

Figure 4-3 : Nouveau paramétrage de la fonction de montaison

De même que la première modification (Kvs) on change aussi Krs en le faisant passer de 0.001 à 0.0005 afin de diminuer le flux sénescent de RV vers RS.

Après cela on s'est rendu compte qu'après la fin de la montaison, la biomasse du compartiment VV augmente très vite grâce au faible flux sénescent. On a donc changé le calcul du flux sénescent svs dans la classe Compartiment_VS afin de faire en sorte qu'il augmente davantage après la fin de la montaison, période avec beaucoup de feuilles âgées car la montaison a limité la production de feuilles au profit des épis. De plus on a toujours un problème au printemps avec le compartiment VV, on a donc dans le même temps ajouté une condition pour que le flux svs soit diminué avant la montaison, période de pleine croissance avec production de nouvelles feuilles. Ce qui nous a donné le calcul suivant pour svs :

Si $T > T1$ et $ST < ST2$ alors

$Kvs = Kvs \times 0.5$

Sinon Si $ST > ST3$ et $T < T4$ alors

$Kvs = Kvs \times 1.5$

$Svs = (1-\sigma) \times Kvs \times T \times VV0$

Désormais le problème c'est la quantité de biomasse du compartiment VV au printemps en effet celle-ci est trop faible, on a donc repoussé le début de la montaison pour laisser plus de temps pour la pousse du compartiment VV, ainsi ST2 passe de 550°C à 600°C.

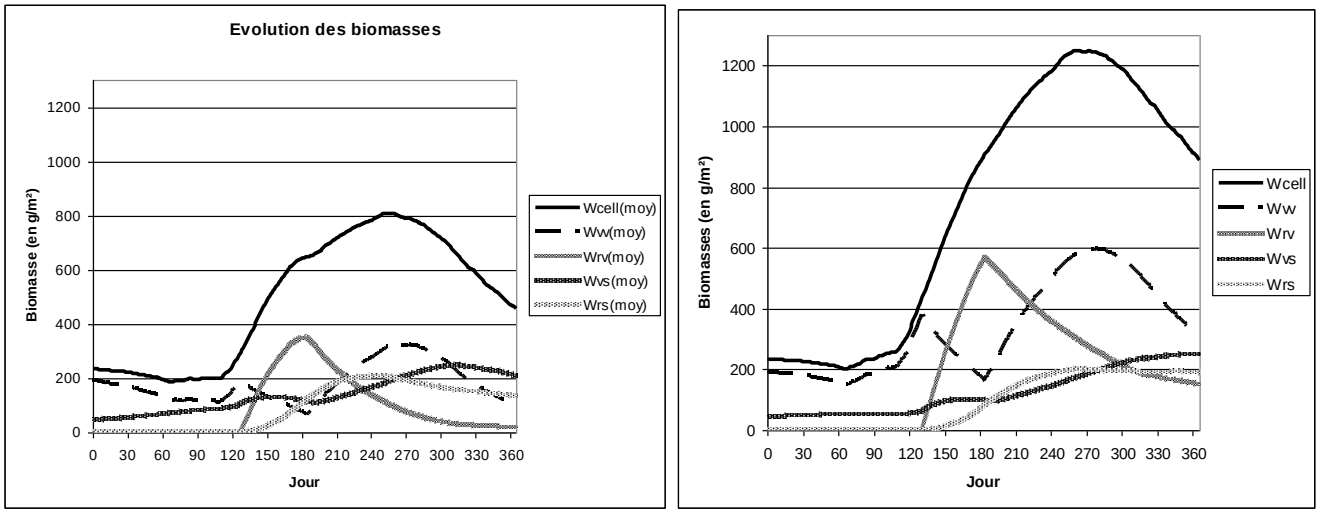

Figure 4-4 : Evolution des biomasses des compartiments au cours d'une année avant le calibrage (à gauche) et après (à droite)

Finalement nous avons abouti au fichier facies.csv qui est présenté en annexe (Annexe 4) qui donne des résultats satisfaisant en terme de simulation du fonctionnement de la végétation.

4.2.2 Pour les modèles animal et social

Dans le cas du modèle animal, les paramètres sont fournis par le fichier *ruminant.csv*, il contient une ligne pour chaque animal du troupeau (voir Annexe 6 pour plus d'informations sur les paramètres). Pour ce qui est de la fonction de distance, c'est-à-dire la fonction qui va permettre au troupeau de rester grégaire, j'ai pris la fonction barycentre avec tout le reste du troupeau.

Pour le modèle social les paramètres sont fournis par le fichier *troupeau.cfg* (voir Annexe 6 pour plus d'informations sur ces paramètres). Parmi tous les paramètres définis dans ce fichier j'ai changé la valeur standard de POURCENTSATISFACTION qui était à 0.8 au début de mon stage or ce paramètre sert à définir la probabilité que l'animal ne soit pas satisfait de la végétation qu'il a sous les yeux et veuille ainsi changer d'endroit en essayant de devenir leader. Mais comme on a pu le voir précédemment cette partie du simulateur n'est pas encore implémentée. Ainsi la valeur 0.8 de ce paramètre autorisé le leadership provoquant ainsi des erreurs au cours de certaines simulations. J'ai mis ce paramètre à 1 pour ne pas avoir de leadership, en effet avec cette valeur l'animal sera toujours satisfait.

J'ai eu aussi à rectifier les valeurs de TEMPSMIN et TEMPSMAX qui étaient respectivement à 9 et 40 secondes, représentant l'intervalle des durées de déplacement entre deux cellules en déplacement tête basse (c'est-à-dire lorsque l'animal va de sa cellule courante à une des 15 cellules de son champ de vision). Or ces temps exagérément grands faisaient que l'animal mettait trop de temps à se déplacer au cours d'une séquence d'ingestion ce qui diminuait énormément les quantités ingérées (l'animal mettait en moyenne 24.5 secondes pour parcourir 1 mètre). Je les ai donc paramétré à 2 et 3 secondes à dire d'expert (voir Annexe 5) afin de se rapprocher des déplacements observés expérimentalement. La vitesse de déplacement est maintenant de l'ordre de 0.4 $m.s^{-1}$, avec des tailles de cellules de 0.1 m² on a une distance entre deux cellules contiguës de 0.34 m donc la durée moyenne pour passer à la cellule suivante est de 0.85 s.

A l'issue de ces étapes de vérification et d'améliorations, nous disposons d'un « prototype 2003 » du simulateur opérationnel, il sera utilisé dans la suite de mon travail. Il est à souligner que l'ensemble de ces améliorations ont été transférées et intégrées dans la version « prototype 2004 » développée par R. Martin [1].

5 Validation

5.1 Qu'est ce que la validation : concepts généraux et techniques

Au début de mon stage, j'ai réalisé avec Michael Gaurut une étude bibliographique en vue du travail de validation que nous allions tous les deux fournir au cours du stage. Parmi toutes les méthodes et techniques que nous avons évoqué dans le document bibliographique (Annexe 5) je n'ai utilisé que des procédés de « validation par comparaison » (Annexe 5 5.b), « validation graphique » (Annexe 5 5.d) le tout soutenu par les « dires d'experts ».

5.2 Outils spécifiques développés pour la validation

En vue de la validation du simulateur, j'ai développé plusieurs outils afin de simplifier le lancement de nombreuses simulations et de traiter rapidement les fichiers de sortie.

5.2.1 Outils d'exécution automatique d'une liste de simulation

Dans le cadre de mon stage j'ai eu rapidement un grand nombre de simulation à lancer avec beaucoup de points communs qui formaient des lots de simulations quasiment identiques. J'ai réalisé un utilitaire en parallèle du simulateur afin de lancer ces simulations sans avoir à passer à chaque fois par l'interface en rentrant toujours les mêmes données. Les simulations prenant souvent pas mal de temps et étant nombreuses il s'est vite révélé intéressant de les lancer la nuit, ce que permet très facilement cet outil.

Pour lancer l'utilitaire il suffit de taper la ligne de commande suivante dans le terminal :

```
./auto.exe 1 chemin1 chemin2 chemin3 chemin4
```

Où `chemin1` correspond au chemin d'accès au fichier contenant les informations des simulations à lancer, `chemin2` correspond au chemin des fichiers d'entrée qui vont servir aux simulations, `chemin3` est le chemin du répertoire de sortie des simulations et `chemin4` est le chemin de l'exécutable du simulateur. La commande suivante permet d'obtenir de l'aide sur le simulateur : `./auto.exe 4`

Lors de son exécution l'outil affiche le suivi des opérations qu'il opère dans le terminal et redirige tous les affichages des simulations dans un fichier *suivit.txt* contenu dans le répertoire de la simulation.

Pour ce qui est des fichiers en entrée, le fichier d'information passé en premier paramètre correspond aux données fournies à l'interface. Elles sont synthétisées sous forme d'un tableau dont les éléments sont séparés par des points virgules, chaque ligne correspondant à une simulation à lancer. Ce fichier contient une ligne de titre, qui n'est pas lue comme une simulation. Attention, le premier élément qui est le nom de la simulation peut contenir des répertoires afin de placer toutes les simulations à des endroits différents pour les classer directement. Par exemple :

(Ligne 2) : 1_animal/chargement_faible ;…
(Ligne 3) : 1_animal/chargement_fort ;…
(Ligne 4) : 5_animaux/chargement_faible ;…
(Ligne 5) : 5_animaux/chargement_fort ;…

Ceci permet de distinguer toutes les simulations par leur type sans avoir toutes les simulations dans le même répertoire avec des noms trop long. Cela est utile lorsque l'on a un très grand nombre de simulations.

Après les premières utilisations de l'outil il s'est avéré nécessaire d'effacer aussi certains des fichiers de sortie des simulations, car ils remplissent très vite tout l'espace disponible. En effet, certains fichiers comme *choix.csv* peuvent atteindre 1 à 1.5 Go en fonction de la taille de la parcelle et de la durée de la simulation. Multiplié par le nombre de simulation on arrive rapidement 40 à 50 Go,, ce qui était l'espace mémoire dont je disposais. J'ai donc supprimé tous les fichiers dont je ne me servais pas à la fin de chaque simulation. Pour donner une idée, il m'est arrivé que mes simulations me prennent 47 Go avec tous les fichiers alors que les fichiers qui m'étaient utiles ne prenaient que 1.5 Go !

5.2.2 Macros pour l'exploitation des sorties du simulateur

Lorsque j'ai commencé à avoir un grand nombre de simulations a traitées, elles mêmes contenant un grand nombre de sorties à traitées puis analysées, j'ai du songé à trouvé une méthode rapide pour ces opérations. Plusieurs solutions se sont présentées à moi malheureusement je n'eu pas suffisamment de temps pour réaliser une étude approfondie sur chacune des méthodes envisagées. Dans les solutions envisagées il y avait l'utilisation de SAS que j'ai écarté puisque je n'avais pas le temps d'aller suffisamment loin dans la validation pour utiliser les outils statistiques qu'aurai apporté SAS. Ensuite il y avait Access, Excel ou Programmer moi même un utilitaire, j'ai choisi de faire des macros sous Excel car je travaillait déjà sous Excel depuis le début du stage ainsi je commençait à me familiariser avec ce logiciel donc en choisissant Excel je pensais gagner du temps. De plus Excel contient déjà les fonctions statistiques de base nécessaire pour la validation que j'avais à faire. Ainsi le choix d'Excel comme outil de traitement des sorties du simulateur m'a permis de faire toutes mes synthèses et analyses des sorties avec ce logiciel.

5.3 Simulations réalisées

5.3.1 Premiers tests avec plusieurs animaux

Les premières simulations que j'ai lancées sur un troupeau avaient pour objectif de vérifier le fonctionnement global du simulateur dans des conditions qui n'avaient pas pu être testées auparavant. C'est-à-dire que j'ai recherché ici quelques éventuels bogues lors de simulations avec troupeau qui auraient pu également apparaître après les modifications que j'ai apportées durant mon stage. Globalement je n'ai donc constaté aucun bogue mais dans cette phase je n'ai pas lancé de simulations dont la durée excéde 20 jours.

5.3.2 Détermination du nombre de réplications approprié

Avant de lancer les premières simulations avec plusieurs animaux, j'ai due déterminer le nombre de réplications nécessaires pour obtenir des résultats significatifs. En effet, lors des étapes de validation précédentes, le prototype ne fonctionnait qu'avec un seul animal, et nous ne savions pas si le niveau de stabilité des sorties serait le même avec un groupe d'animaux. En première approche, j'ai réalisé deux simulations, une de cinq réplications et l'autre d'une seule mais. J'ai utilisé deux graines différentes pour le générateur de nombres aléatoires afin de garantir l'indépendance des résultats (Figure 5-1).

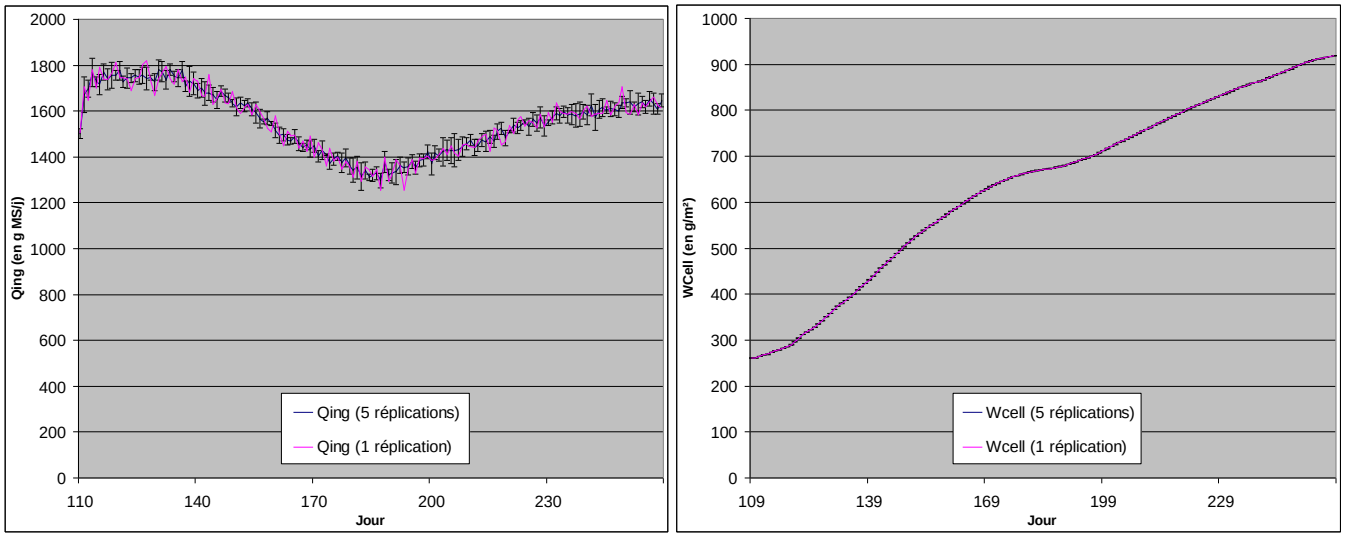

Figure 5-1 : Comparaison des résultats entre 1 réplication et 5 réplications

On constate clairement sur ces deux graphiques que le nombre de réplications n'a aucune influence sur la végétation et une influence minime sur la quantité ingérée. Nous avons donc fait le choix compte tenu de nos objectifs de ne travailler qu'avec une seule réplication.

5.3.3 Vérification de la grégarité du troupeau

Afin de vérifier le fonctionnement de la relation sociale « grégarité », j'ai analysé la répartition spatiale du troupeau. J'ai lancé deux simulations l'une avec la fonction « min » (Figure 5-2) et l'autre avec la fonction « barycentre » par rapport à tout le troupeau (Figure 5-4).

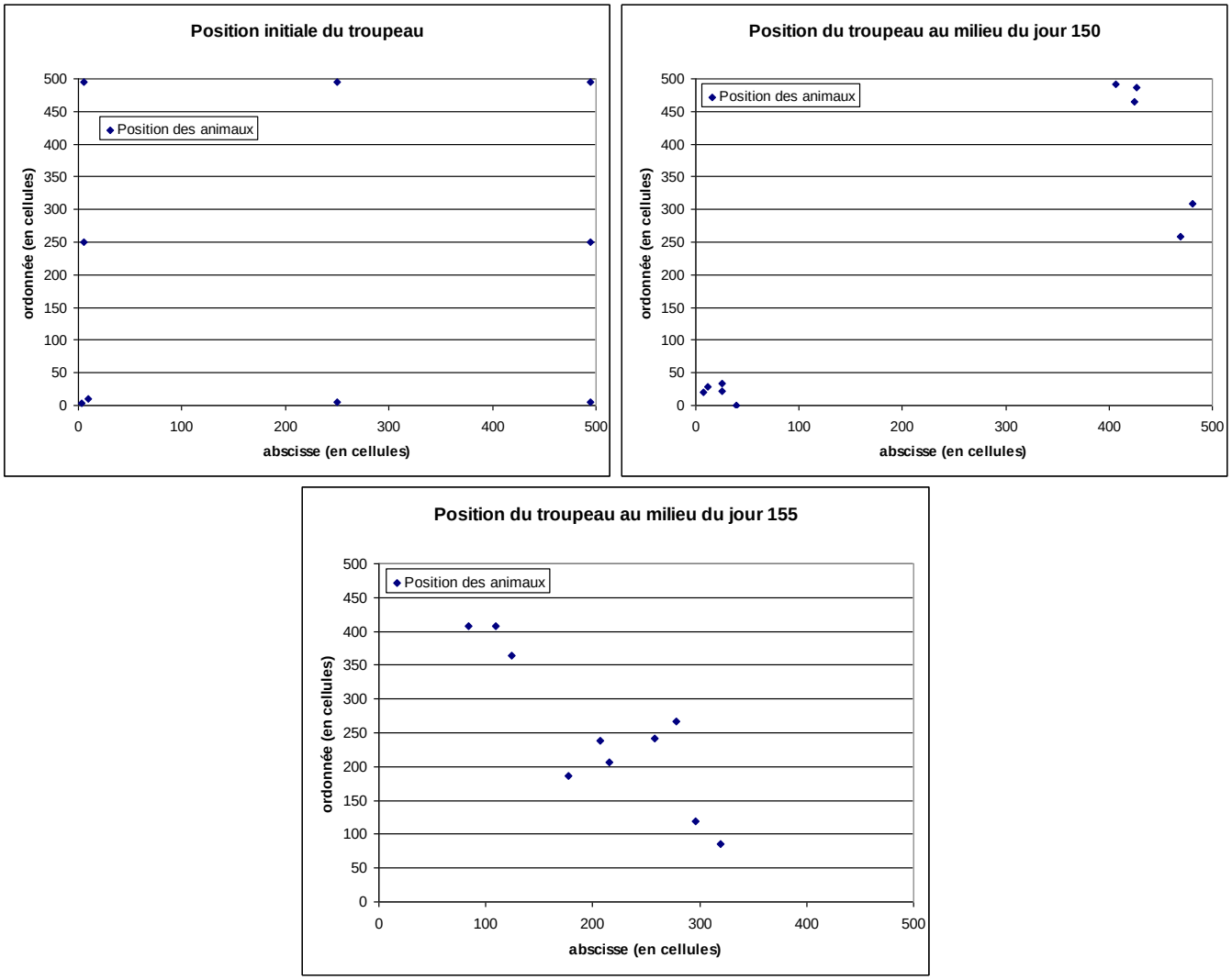

Figure 5-2 : Simulation avec la fonction de distance « min »

Dans le cas de la fonction « min » on peut constater que l'on n'a qu'une faible impression de grégarité voir la formation de plusieurs groupes. La grégarité n'est pas suffisante ce qui se comprend facilement puisque chaque animal n'est lié qu'à un seul autre animal du troupeau ce qui peut aboutir à des situations problématiques comme l'alignement des animaux en rang qui ne forment plus alors un troupeau grégaire (Figure 5-3).

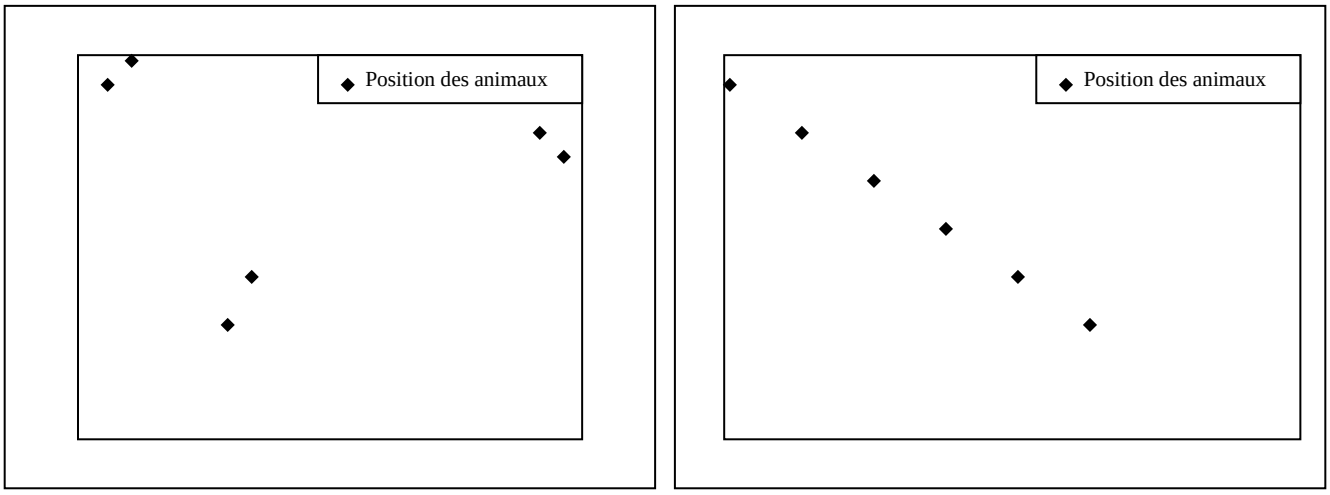

Figure 5-3 : Positionnements non grégaires possibles avec la fonction « min »

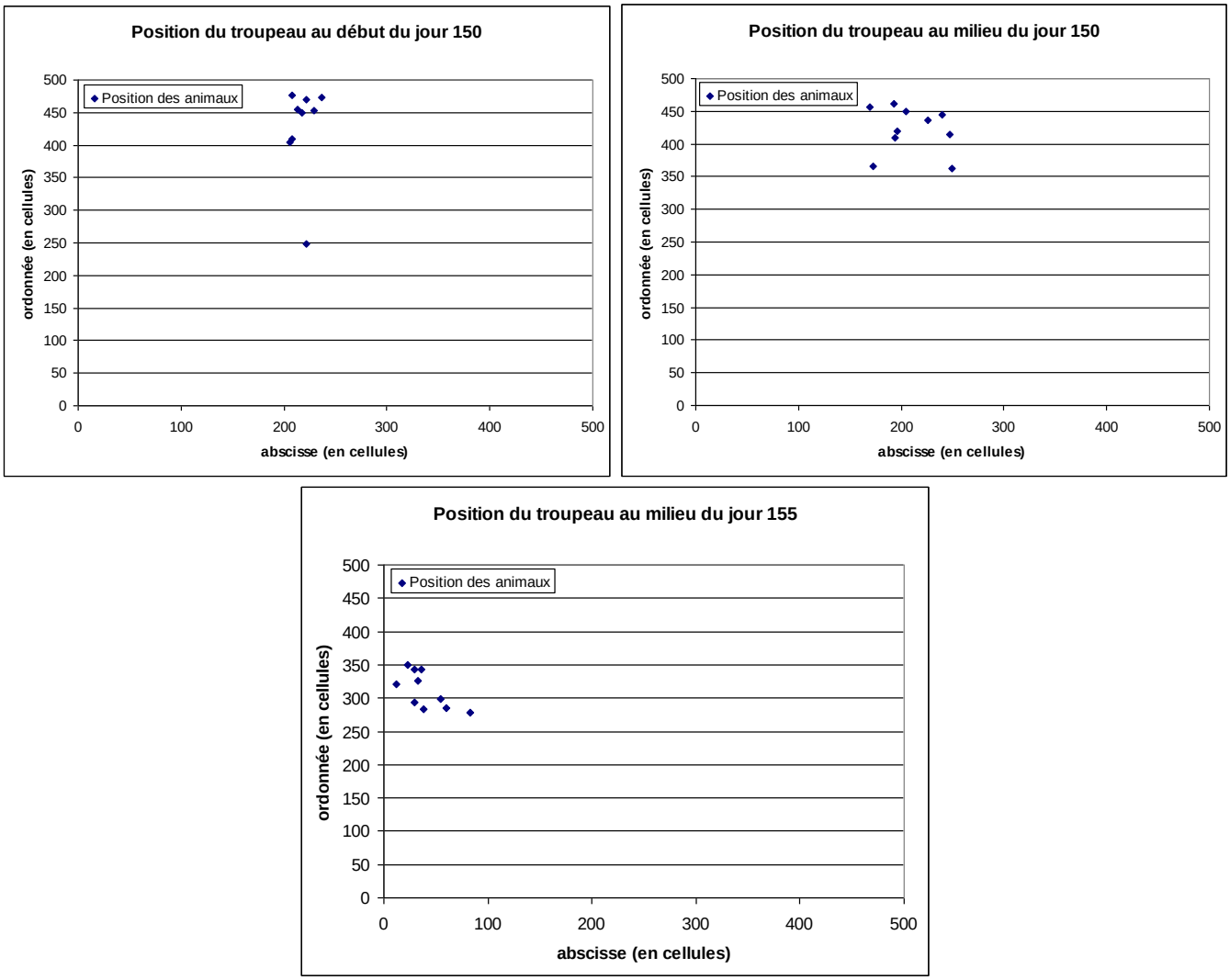

Figure 5-4 : Simulation avec la fonction de distance « barycentre » par rapport à tout le troupeau avec les mêmes positions de départ que la simulation avec la fonction « min » (Figure 5-2)

Dans le cas de la fonction « barycentre », on observe un troupeau plus compact, les animaux ne s'éparpillant pas sur toute la parcelle.

On en conclu que les fonctions de distance vont permettre d'échelonner la grégarité du troupeau par le paramètre « NbProches » de la fonction « barycentre ». En effet, si ce paramètre vaut 1 on retombe dans le cas de la fonction « min ». Dans la suite du travail nous avons toujours utilisé la fonction « barycentre ».

5.3.4 Etude du chargement

Les objectifs de cette partie sont d'une part d'étudier le comportement du simulateur lorsque le chargement de la parcelle augmente et d'autre part de comparer les résultats obtenus avec deux méthodes de variation du chargement différentes. En effet, dans le simulateur PARIS on peut faire varier le chargement de la parcelle soit en jouant sur le nombre d'animaux par unité de surface fixe, soit en jouant sur la taille de la parcelle à nombre d'animaux constant. Nous étions intéresser de tester si pour un même niveau de chargement, nous obtenions les même résultats suivant la façon dont nous avions défini le niveau de chargement (taille groupe, ou taille parcelle). Pour ce faire nous avons conçu le protocole expérimental suivant :

On fait d'abord varier la surface de la parcelle pour un nombre d'animaux constant :

Surface (en m²)	**750**	**1500**	**3000**	**6000**
Nombre d'animaux	5	5	5	5
Chargement (en brebis/ha)	66.7	33.3	16.7	8.3

Ensuite on fait varier le nombre d'animaux à surface constante :

Surface (en m²)	3600	3600	3600	3600
Nombre d'animaux	**3**	**6**	**12**	**24**
Chargement (en brebis/ha)	8.3	16.7	33.3	66.7

5.3.4.1 Problème d'arrêts prématurés des simulations

Lors de ces simulations j'ai rencontré deux bogues qui ont engendrés des arrêts prématurés de simulations empêchant ainsi une étude complète de leurs résultats. Ainsi dans la simulation avec un chargement de 66.7 brebis/ha, le prélèvement du troupeau est nettement supérieur à la croissance de l'herbe. En conséquence, il arrive un moment où il n'y a plus suffisamment d'herbe sur la parcelle et le comportement des animaux devient aberrant. Le simulateur n'ayant pas été prévu pour ces situations, on observe que certaines variables du modèle animal prennent alors des valeurs aberrantes, comme par exemple le compartiment SP (petites particules dans le rumen) qui devient négatif. Un certain nombre de tests de cohérence devront être rajoutés pour éviter ces problèmes.

J'ai également observé des arrêts prématurés des simulations dans des situations où le chargement est moins élevé, et où le fonctionnement de la végétation et de l'animal semble normal. Je n'ai pas encore trouvé d'explication à ces arrêts, il semble qu'ils interviennent d'autant plus précocement que le nombre d'animaux est élevé dans la simulation. Il faudra donc rapidement trouver et résoudre cette erreur, car le simulateur a été conçu pour pouvoir fonctionner avec un nombre d'animaux élevé pendant de longues périodes.

5.3.4.2 Analyse des effets du chargement

J'ai choisi d'analyser ces effets sur les résultats obtenus avec la méthode de chargement par augmentation de la surface de la parcelle.

5.3.4.2.1 Sur la végétation

Dans un premier temps j'ai analysé l'évolution des biomasses de la parcelle (Figure 5-5). Tout d'abord on peut regarder le cas du plus faible chargement (8.3 brebis/ha), c'est la simulation avec un troupeau pâturant la parcelle qui est la plus proche de la situation sans troupeau (parmi le protocole expérimental) (Figure 5-6), on peut donc comparer ces deux situations. On constate ainsi une évolution au cours de la saison (jours 109 à 261, du 18 Avril au 17 Septembre) quasiment identique dans ces deux cas, la seule différence se situe au niveau de la quantité de biomasse présente dans chaque compartiment qui est inférieure lorsque les animaux sont présents (dû aux défoliations de ces derniers). Ensuite, si on augmente le chargement à 16.7 brebis/ha on maintient encore une évolution similaire à la végétation seule mais avec cette fois ci nettement moins de biomasses. En passant au chargement supérieur (33.3 brebis/ha), on commence à apercevoir sérieusement la perturbation apportée par le troupeau sur la végétation. En effet, la biomasse totale (WCell) a désormais du mal à augmenter et n'atteint plus que 420 g/m² au jour 229 (16 août) au lieu de 800 g/m² pour 16.7 brebis/m². Enfin lors d'un chargement très élevé (66.7 brebis/ha), le troupeau prend le dessus sur la végétation et fini par manger toute la biomasse présente vers le jour 160.

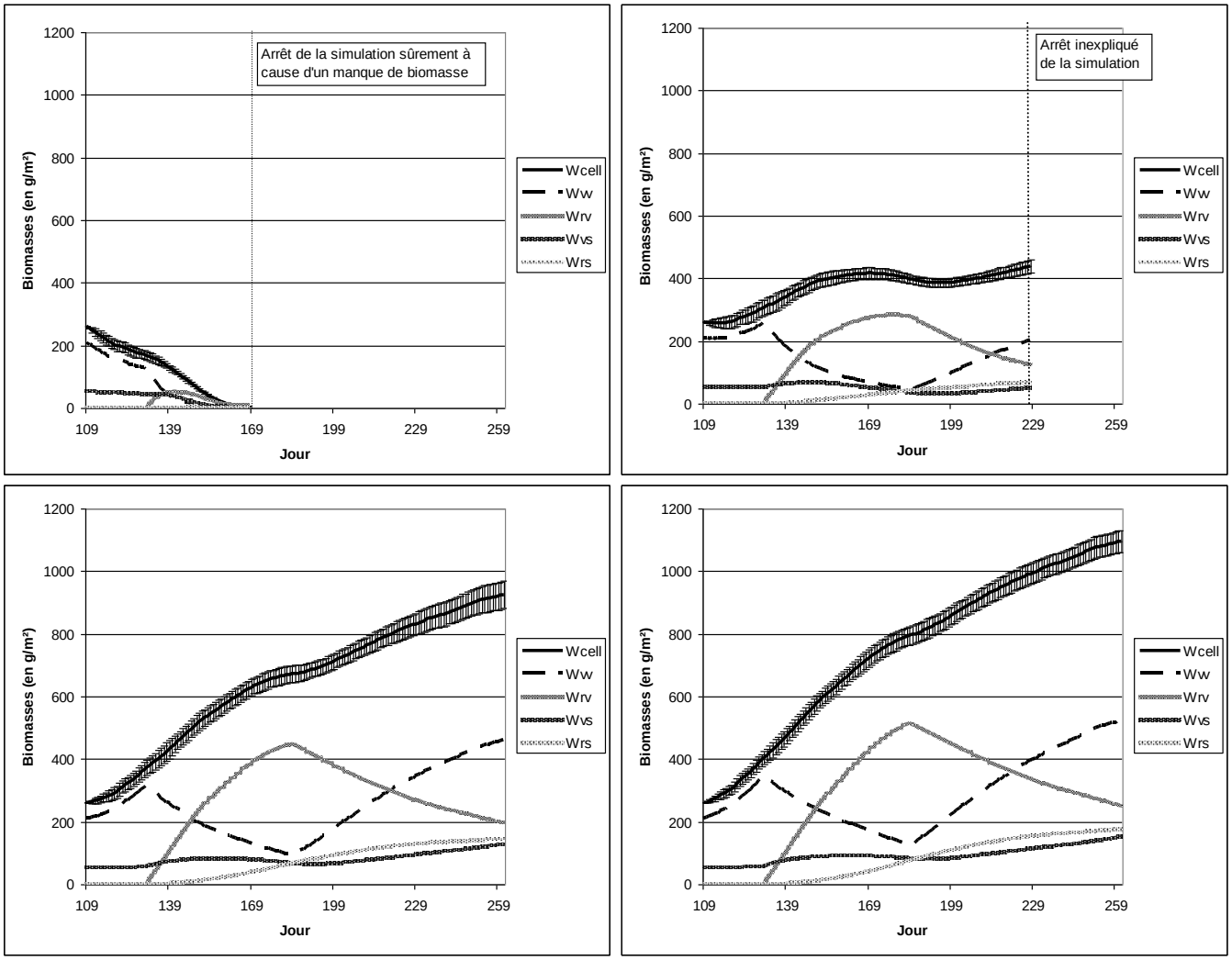

Figure 5-5 : Evolution des biomasses pour les quatre chargements 66.7, 33.3, 16.7 et 8.3 brebis/ha

Par la suite je me suis intéressé plus en détail à l'évolution de la biomasse Wrv car celle-ci avait fait l'objet d'une modification qui consistait à prendre en compte le contrôle de la montaison par l'animal, c'est-à-dire que lorsqu'il mange trop de RV sur une cellule donnée durant la période de montaison alors celle-ci doit arrêter sa montaison. Or cette modification devrait avoir pour effet de beaucoup diminuer la biomasse du compartiment RV. Malheureusement en regardant les courbes des trois plus faibles chargements j'ai constaté que ce dernier ne diminuait pas suffisamment car vers la fin de la montaison (jour 180) la biomasse de ce compartiment représente les deux tiers de la biomasse totale.

Enfin j'ai représenté sur ces quatre graphiques pour la courbe de WCell les écart-types permettant de se faire une idée de l'évolution de l'hétérogénéité de la parcelle puisque l'écart-type augmente dans les chargements les plus faibles.

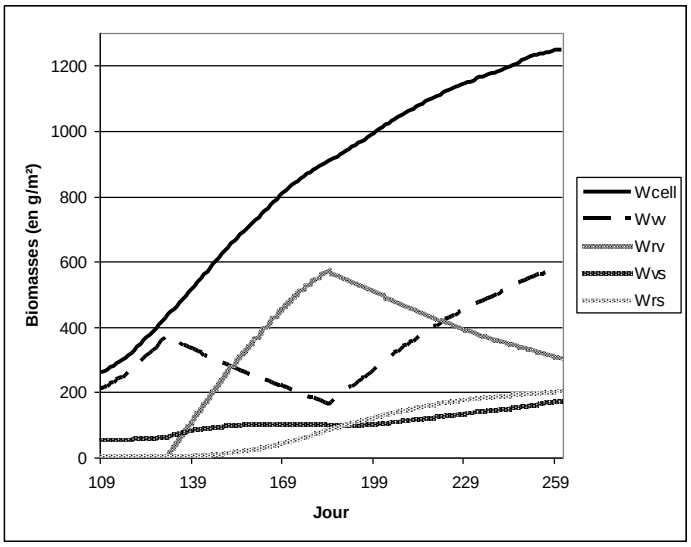

Figure 5-6 : Evolution des biomasses de la parcelle pendant la saison étudiée ici

5.3.4.2.2 Sur le troupeau

Ensuite j'ai regardé l'évolution de deux variables d'ingestion des animaux, la quantité ingérée cumulée par journée et la durée d'ingestion cumulée par journée qui sont moyennées sur tout le troupeau. Dans les cas des chargements 8.3 et 16.7 brebis/ha on ne constate pas de différence significative ce qui est cohérent puisque les animaux mangent dans ces deux cas suffisamment de nourriture de qualité. Alors que dans le cas 33.3 brebis/ha et encore plus dans celui à 66.7 brebis/ha, les animaux sont obligés de compenser la baisse de qualité et de quantité de biomasse de la végétation (dû à une défoliation excessive car le chargement est trop fort) qui diminue les quantités ingérées par les animaux, par une augmentation de leurs durées d'ingestion pendant la journée. C'est-à-dire qu'un animal est obliger de remplacer une défoliation par plusieurs défoliations de moindre valeur nutritive afin de satisfaire ses besoins alimentaires. Ainsi dans le cas extrême, le troupeau mange normalement jusqu'au jour 132 puis il est obligé de compensé en augmentant sa durée d'ingestion jusqu'à sa durée maximale, toute la journée (1440 minutes).

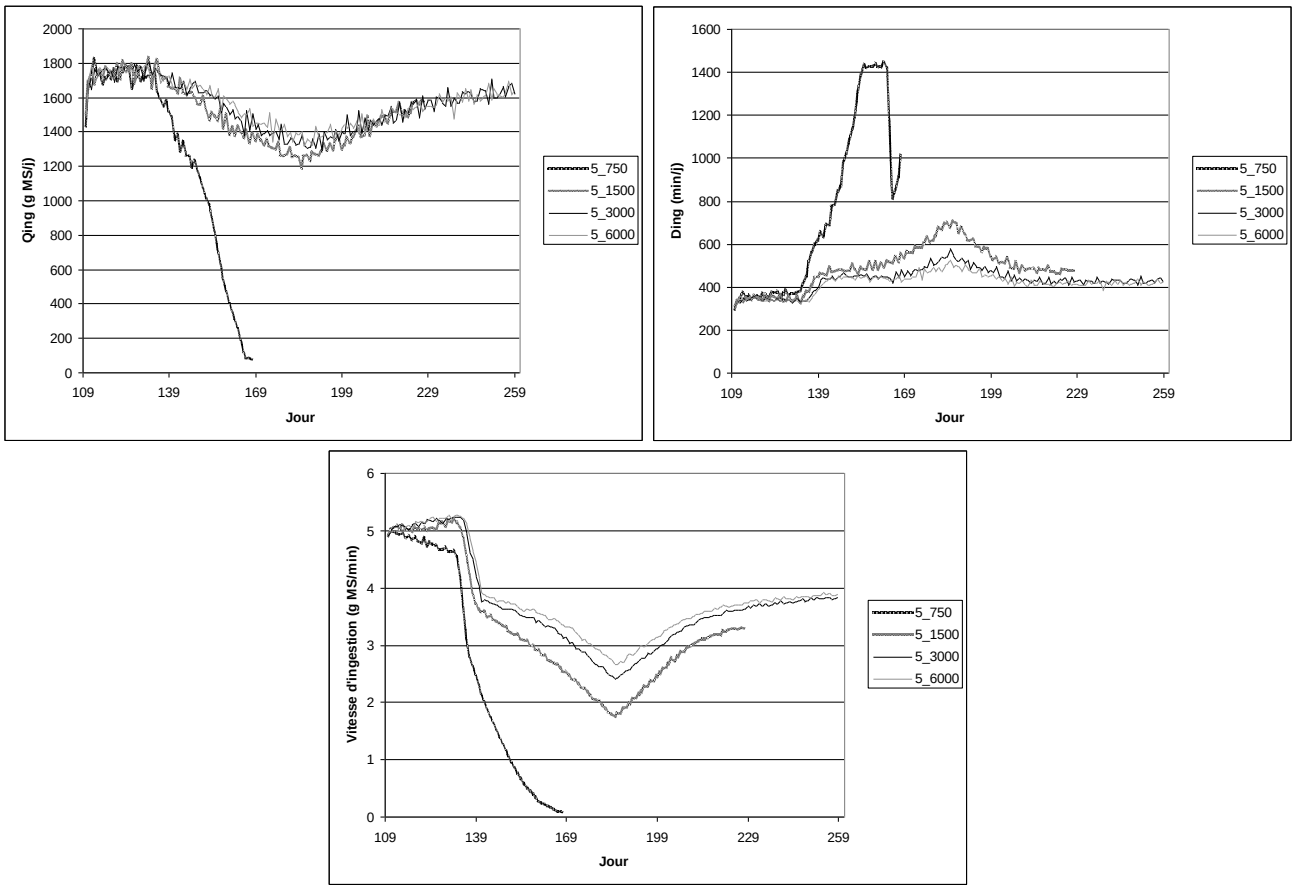

Figure 5-7 : Evolution de la moyenne des quantités ingérées par jour de chaque animal, de leurs durées d'ingestion et de leurs vitesse d'ingestion

Concernant la vitesse d'ingestion (modifiée durant mon stage, voir 4.1.1) on retrouve ici les différentes équations utilisées. En effet, jusqu'au jour 132 on est avant la montaison, ensuite du jour 132 au jour 184 il y a la montaison et enfin la troisième phase se fait sans elle, or sur le graphique on distingue quatre phases parmi lesquelles on a les trois précédente et la dernière correspond à la transition que j'ai implantées. Mais on peut remarquer que contrairement à ce l'on voulait cette transition est encore un peu brutale peut-être faudrait-il la revoir afin de faire en sorte qu'elle s'étale plus dans le temps, ou alors cela est du à un mauvais contrôle du compartiment RV (voir 5.3.4.3).

5.3.4.3 Comparaison des deux méthodes de chargement

En analysant maintenant les courbes obtenues pour une surface constante (Annexe 8), on retrouve encore des erreurs aboutissant à la fin prématurée de la simulation mais le début des simulations permet quant même une comparaison. Afin de repérer les différences dans l'évolution des biomasses j'ai réalisé des superpositions des courbes des deux méthodes respectives pour les quatre chargements. La conclusion fût sans équivoque, les courbes se superposant parfaitement (il en va de même pour les courbes animales). Ainsi on vérifie ici un point important du comportement du simulateur, en effet il aurait été totalement anormal que deux types de chargement différents donnent des résultats très différents.

5.3.5 Vérification du contrôle de la montaison

Parmi les modifications apportées au simulateur sur le « prototype 2003 » il y a l'ajout du contrôle de la montaison par les défoliations de l'animal (4.1.2). Cet ajout mérite un intérêt

particulier ici car il n'a pas vraiment encore été testé avec des animaux. Ainsi cette partie a deux objectifs :

- comparer les résultats fournis avec chacune des deux méthodes étudiées (Dans tout le travail précédent de mon stage c'est la seconde méthode qui a été utilisée). La première fonctionne par rapport à la hauteur des épis, dans cette méthode on considère que la montaison s'arrête si l'animal prélève presque toute la hauteur. La seconde méthode se base sur la probabilité que l'animal prélève la quasi-totalité des épis de la cellule (voir une description plus détaillées dans le rapport de Martin et Pean [9]).
- analyser les résultats donnés par chacune pour s'assurer de leurs fonctionnements. Pour cela je vais regarder l'évolution de la biomasse du compartiment RV et la variabilité de cette biomasse entre les cellules de la parcelle.

A l'issue du lancement des simulations un premier constat s'imposant, il y a un manque de sécurité autour de la seconde méthode car 2 des 3 simulations lancées avec celle-ci s'arrêtent au bout de quelques jours car certains compartiments RV deviennent négatifs. Ce phénomène peut être du à un mauvais agencement des évènements de mise à jour de parcelle, en effet on calcul d'abord l'évolution des biomasses des compartiments avant de regarder s'il y a eu un contrôle de montaison et donc un arrêt de la croissance du compartiment RV. Ainsi avec les défoliations du troupeau il peut arriver que l'on fasse passer une quantité de biomasse de RV vers RS trop grande provoquant un passage à une biomasse négative du compartiment RV.

Ainsi les seules simulations qui furent comparables sont celles avec un animal sur 6000 cellules. En analysant l'évolution de la biomasse (Figure 5-8), on constate que celle-ci augmente nettement moins avec la seconde méthode (300 g/m² au maximum au lieu de 450 g/m² pour la première méthode). L'interprétation de cette différence ne peut être liée qu'au contrôle de la montaison qui est plus stricte avec la seconde méthode.

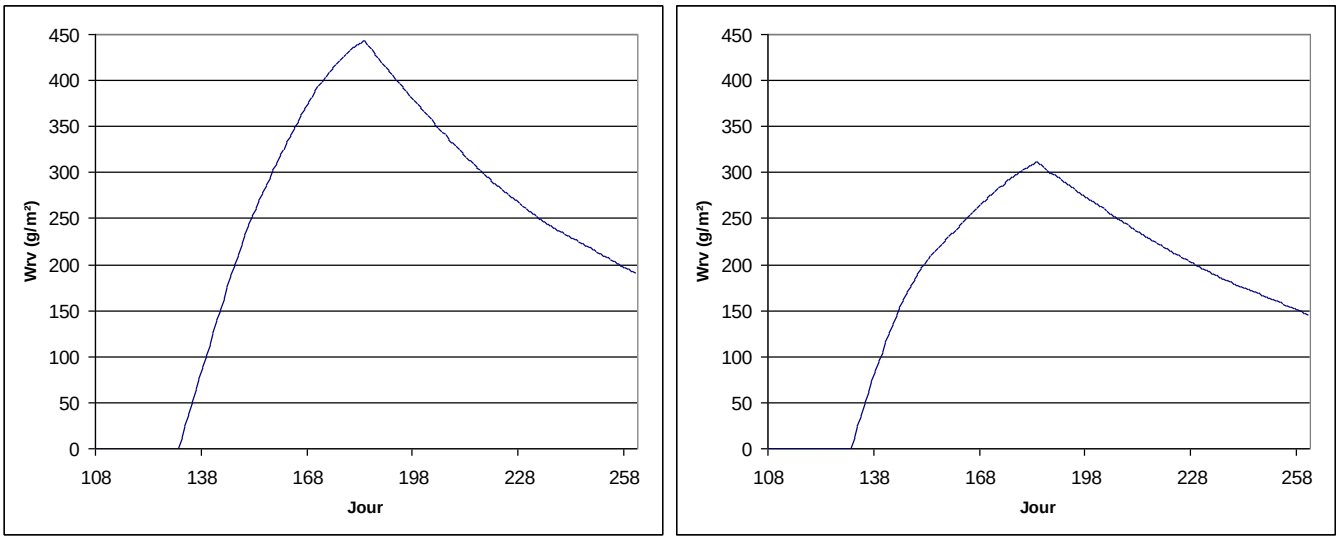

Figure 5-8 : Evolution de la biomasse du compartiment RV pour la deuxième méthode (à gauche) et la troisième (à droite)

Maintenant si on regarde la variabilité de la biomasse de RV entre les cellules (Figure 5-9), on constate qu'elle augmente nettement plus avec la seconde méthode ce qui implique donc qu'il y a beaucoup plus de cellules contrôlées par les défoliations du troupeau. C'est-à-dire qu'une grande partie des cellules ont une biomasse de RV quasiment nulle du fait de ce contrôle qui a stoppé la croissance de ce compartiment (montaison).

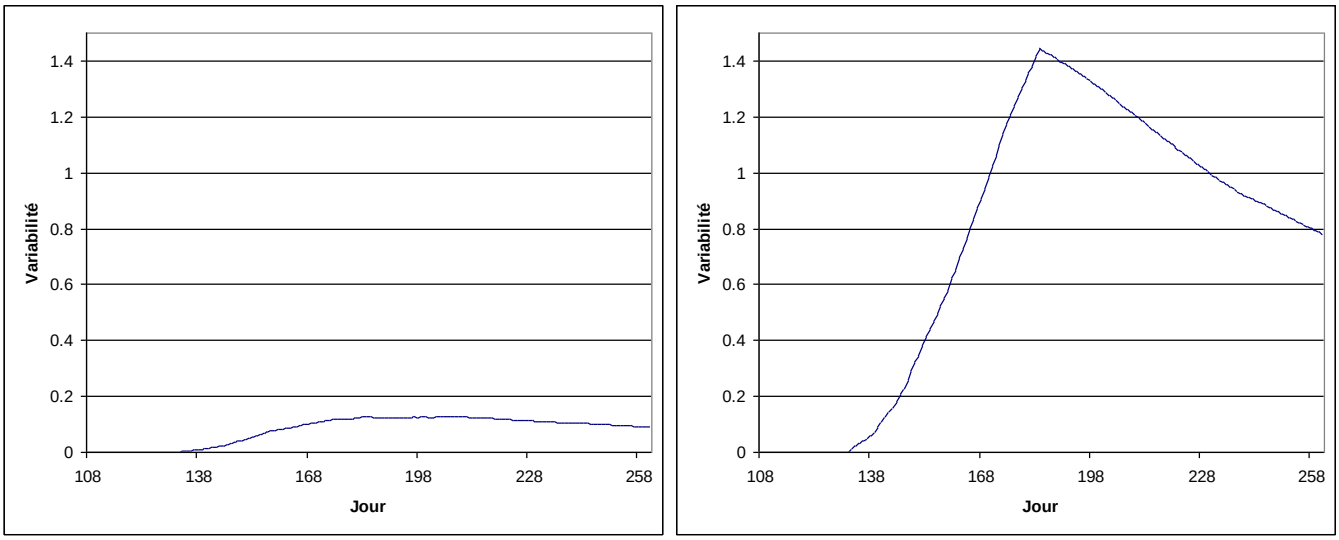

Figure 5-9 : Evolution de la variabilité de la biomasse du compartiment RV entre les cellules de la méthode 1 (à gauche) et la méthode 2 (à droite)

Ainsi après cette étude on peut dire que la seconde méthode est bien plus efficace puisqu'elle fait apparaître une très grande diversité entre les cellules de la parcelle prouvant l'intervention du contrôle ce qui n'est pas le cas avec l'autre méthode. A l'avenir il pourra s'avérer utile de refaire une partie de mon analyse avec la troisième méthode puisqu'elle fonctionne mieux (en effet la méthode utilisée dans mon analyse était la deuxième).

6 Discussion

6.1 Contraintes techniques

Pendant la première période de mon stage, j'ai dû faire face à un certain nombre de problèmes techniques qui ont engendrés une perte de temps et un manque d'efficacité durant cette période. En effet, j'ai dû me passer de l'utilisation de l'ordinateur les deux premières semaines car aucun compte n'avait été configuré pour moi. De plus le responsable des ressources informatiques qui aurait pu résoudre ce problème était en vacance et j'ai dû attendre son retour pour avoir mon compte. Ce contretemps a repoussé ma première prise en main du simulateur ce qui a rendu par moment un peu trop abstrait ce que je lisais pour ma bibliographie sur le simulateur.

Par la suite un deuxième problème apparu lors de l'installation de LINUX sur mon ordinateur qui était nécessaire car le simulateur a été développé pour fonctionner sous ce système d'exploitation. Mais vu que les responsables de mon stage avaient décidés d'acheter un ordinateur récent, pour réduire le temps de simulation, les drivers LINUX pour certains des matériels n'existent pas, parmi ceux-ci il y avait le disque dur qu'il a donc fallu changer.

Enfin, un dernier problème demeura, lorsque j'ai pu lancer mes premières simulations j'ai testé le temps d'exécution des grosses simulations et il s'avéra que l'ordinateur redémarrait au bout de 4 à 5 minutes de simulation. Cela était du à un mauvais refroidissement du processeur qui dépassait alors sa température maximale (78°C). On a donc changé le ventilateur du processeur afin dans mettre un plus puissant mais j'ai dû attendre plus de deux semaines avant que le fournisseur ai reçu le ventilateur et vienne l'installer sur mon ordinateur. Ainsi pendant cette durée j'ai du me contenter de très courtes simulations lors de mes tests.

6.2 Forces et faiblesses du simulateur

6.2.1 Résultats des temps d'exécution

En même temps que j'appliquais les modifications de fonctionnement du simulateur, j'ai aussi fait tout mon possible pour optimiser la mémoire utilisée et diminuer le temps d'exécution. Mais vu que j'ai rajouté de nombreux traitements (3 fichiers de statistiques et un fichier journalier en plus par rapport au « prototype 2003 »), le temps d'exécution a augmenté. Voici un exemple sur une simulation de 50 jours avec 3 ruminants sur une parcelle de 30000 cellules (la simulation commence avec la végétation seule au jour 1 = 1er Janvier) :

« Prototype 2003 » : 2 minutes et 25 secondes

« Prototype 2003 bis » : 4 minutes et 53 secondes

Il faut tout de même bien garder à l'esprit que le traitement le plus long est ici les écritures dans les fichiers de sortie. En effet dans l'état actuel du simulateur tous les fichiers sont créés, ainsi en mettant un choix des fichiers de sortie dans l'interface seuls les fichiers souhaités seront écrits permettant de réduire considérablement le temps d'exécution.

6.2.2 Fonctionnement informatique

Mon stage a apporté au simulateur de nombreuses avancées pour améliorer son fonctionnement en terme de réalité biologique et de robustesse du code. Ainsi les fichiers de sortie ont été remodelés pour faciliter leurs utilisations, sur ce point quelques éléments restent

à améliorer (**2.2**). Certains bogues persistent comme les arrêts prématurés constatés lors de la phase de validation (**5.3.4.1**). Enfin certains aspects pratiques sont à revoir dans l'interface comme le fait de pouvoir choisir les fichiers de sortie souhaités afin d'éviter une occupation inutile d'une grande place mémoire du disque dur.

Dans la suite du développement du simulateur PARIS il sera intéressant de reprendre totalement le codage du simulateur afin d'optimiser son fonctionnement. En effet, le code a évolué en lui ajoutant des morceaux au fur et à mesure des stages ce qui ne permet pas de réaliser un code bien optimisé.

6.2.3 Fonctionnement biologique

Au cours de ce stage j'ai du passé la majorité de mon temps en vérification, ainsi l'objectif d'origine du stage qui été la validation du simulateur avec un troupeau d'animaux n'a été qu'amorcé.

Lors du début de validation que j'ai effectué, j'ai étudié de manière superficielle le fonctionnement biologique du simulateur. Désormais il sera intéressant de poursuivre cette étude en rentrant plus dans les détails du fonctionnement de la digestion de l'animal, ses choix de déplacements, sa rumination, etc. De plus avec ce que l'on a vu précédemment il sera utile de regarder plus en détail les chargement de la parcelle compris entre 66.7 brebis/ha où l'on détruit la parcelle et 33.3 brebis/ha où l'on commence à voir une surcharge de la parcelle.

6.3 L'avenir du simulateur

Au cours de mon stage le simulateur PARIS est passé du « prototype 2003 » obtenu par S. Portal au « prototype 2003 bis ». Ce dernier prototype a déjà été réutilisé par R. Martin afin de lui ajouter la mémoire spatiale de l'animal, il a ainsi obtenu le « prototype 2004 » qui sera utilisé pour la suite du développement après que le « prototype 2003 bis » soit totalement validé. Pour ce qui est du prototype que j'ai obtenu il sera aussi utilisé par les biologistes afin de pouvoir continuer à lancer des simulations et à les analyser jusqu'à ce que le « prototype 2004 » aboutisse à une version utilisable. Dans cette optique il sera indispensable de réaliser un mode d'emploi complet pour les biologistes afin qu'ils ne soient pas bloqués par l'utilisation de LINUX lors de l'installation par exemple. Enfin il sera nécessaire de poursuivre le travail que j'ai commencé lors de ce stage, c'est-à-dire la validation du « prototype 2003 bis » afin de fournir une base de travail pleinement opérationnelle aux biologistes, en particulier résoudre le problème de l'arrêt prématuré des simulations avec plusieurs animaux.

Conclusion

L'objectif du stage était de vérifier le « prototype 2003 » de S. Portal puis de commencer à valider le « prototype 2003 bis » que j'ai obtenu après correction et amélioration. J'ai atteint ces objectifs avec des améliorations au niveau des fichiers de sortie, des corrections faisant suite au travail de validation de M. Jouven, d'autres faisant suite à mon travail de vérification. Ensuite la validation que j'ai débutée avec plusieurs animaux a fait apparaître deux bogues qui seront à corriger soit à la fin de mon stage soit ultérieurement, mais elle a aussi montrée que le simulateur répondait assez bien aux variations de chargement.

Au cours de ce stage j'ai rencontré quelques difficultés, notamment dans la compréhension du code du simulateur qui demande beaucoup de temps et nécessite la construction d'un grand nombre de diagrammes UML.

Cependant, ce stage m'a permis de concrétiser et de perfectionner certaines connaissances comme le langage C++. En effet le fait de devoir s'approprier un code que l'on n'a pas programmé et qui est d'un volume assez conséquent nécessite une bonne compréhension de tous les outils et techniques du langage C++ afin de comprendre leurs utilisations.

La validation que j'ai entamée devra être complétée à l'avenir pour offrir des bases saines afin de poursuivre la conception du simulateur. Car il manque encore dans le simulateur la notion de « leadership », les action « boire » et « repos long », et la mémoire quant à elle a été implémentée par R. Martin en parallèle de mon stage.

A terme le simulateur pourra être utilisé pour élargir les connaissances des biologistes dans le domaine de « l'utilisation par le pâturage et l'évolution de la végétation des surfaces herbagères hétérogènes et peu chargées ».

Références bibliographiques

[1] R. Martin, *Analyse et conception du leadership et de l'utilisation de la mémoire dans le simulateur PARIS*, rapport de stage INRA, 2004

[2] F. Guerry, *Conception d'un simulateur multi-agents de l'interaction troupeau/parcelle pâturée*, rapport de stage INRA, 2000

[3] L. Masson, *Modélisation de la mémoire spatiale des animaux dans un simulateur parcelle/troupeau*, rapport de stage INRA, 2001

[4] G. Martin, *Analyse et conception du couplage entre les modules animal et végétal du simulateur troupeau/parcelle*, rapport de stage INRA, 2002

[5] S. Portal, Conception *du module spatial et social d'un simulateur parcelle/troupeau*, rapport de stage INRA, 2003

[6] M. Jouven, *Validation du modèle animal-végétal dans le cadre du développement d'un simulateur parcelle-troupeau*, rapport de stage INRA, 2003

[7] C. Aujas, *Validation d'un modèle stochastique de simulation de l'occurrence des mammites et de leur impact sur la production laitière des vaches*, mémoire CUST-INRA, 1995

[8] C. Pissavy, *Analyse de la sensibilité du module végétation d'un modèle spatial de l'interaction prairie-troupeau*, rapport de stage INRA, 2002

[9] R. Martin & B. Pean, *Simulateur Prairie-Troupeau : Développement d'un module de défoliation contrôlée*, rapport de projet ISIMA / INRA, 2003-2004

[10] Gauch et al., *Comparing model-based prediction and measured values*, agronomy journal vol. 95, november-december 2003, p 1442-1446

[11] R. G. Sargent, *Verification and validation of simulation models*, proceedings of the 1998 winter simulation conference, p 121-130

[12] R. G. Sargent, *Verification and validation of simulation models*, proceedings of the 2003 winter simulation conference, p 37-48

[13] Jack P. C. Kleijnen, *Validation of models : statistical techniques and data availability*, proceedings of the 1999 winter simulation conference, p 647-654

[14] C. Mazel, *La validation et la vérification de modèle de simulation*, non publié

[15] F. Bavaud, *Analyse multivariée*, cours, Université de Lausanne, novembre 2001

[16] J. P. C. Kleijnen, *Experimental designs for sensivity analysis of simulation models*

[17] J. P. C. Kleijnen, David Deflandre, *Validation of regression metamodels in simulation : Bootstrap approach*, 2002

[18] Hugh G. Gauch, J. T. Gene Hwang, Gary W. Fick, *Model evaluation by comparaison of model-based predictions and measured values*, Agronomy journal, vol. 95, novembre-décembre 2003, p 1442-1446

[19] I. Halachmi, A. Dzidic, J. H. M. Metz, L. Speelman, A. A. Dijkhuizen, J. P. C. Kleijnen, *Case study : Validation of simulation model for robotic milking barn design*, European journal of operational research 134, 2001, p 677-688

[20] Ludovic Lebart, *Validation et rééchantillonnage*, Cycle postgrade de statistique, Université de Neuchâtel, 2004

[21] G. Faÿ, *Probabilités et Statistiques*, Licence MIAGE, *Notes de cours*, Université des Sciences et Technologies de Lille, 2003-2004

[22] T. Lemlouma, A. Boubina, *L'intelligence artificielle distribuée et les systèmes multi-agents*, http://opera.inrialpes.fr/people/Tayeb.Lemlouma/Papers/IAD_Presentation.pdf publication INRIA Rhône-Alpes, 2001

[23] B. Dumont, D. R. C. Hill, R. Baumont, P. Carrère, L. Masson, C. Mazel, L. Pérochon, *Modélisation des processus spatiaux d'utilisation de couverts herbacés hétérogènes par les herbivores : Représentation de leur mémoire spatiale*, Acte du séminaire INRA-CIRAD Modélisation du fonctionnement des troupeaux, Millau, pages 25-32, 2001

[24] D. Sauvant, R. Baumont, P. Faverdin, *Development of a Mechanistic Model of Intake and Chewing Activities of Sheep*, Journal of Animal Science, 74, pages 2785-2802, 1996

[25] R. Baumont, D. Cohen-Salmon, S. Prache, D. Sauvant, *A mechanistic model of intake and grazing behavior in sheep integrating sward architecture and animal decisions*, Animal Feed Science and Technologie, 2004

[26] F. Garcia, P. Carrère, J.F. Soussana, R. Baumont, *The ability of sheep at different stocking rates to maintain the quality and quantity of their diet during the grazing season*, journal of Agricultural Science 140, pages 113-124, 2003

[27] E. Begou, C. Descours, *Simulateurs de troupeaux : Construction de données à partir des sorties pour obtenir des statistiques*, rapport de projet ISIMA / INRA, 2003-2004

[28] F. Louault, J.F. Soussana, M. Perrodin, *Long term effect of a reduce herbage use in a semi-natural grassland*. I-Plant functional traits and plant response groups, Grassland Science in Europe Vol 7, pages 338-339, 2002

[29] P. Carrère, E.E. Sosinski Jr., F. Louault, J.F. Soussana, *Validation of a model simulating grassland vegetation dynamics using plant traits measured along a gradient of disturbance*, Grassland Science in Europe Vol 9, pages 784-786, 2004