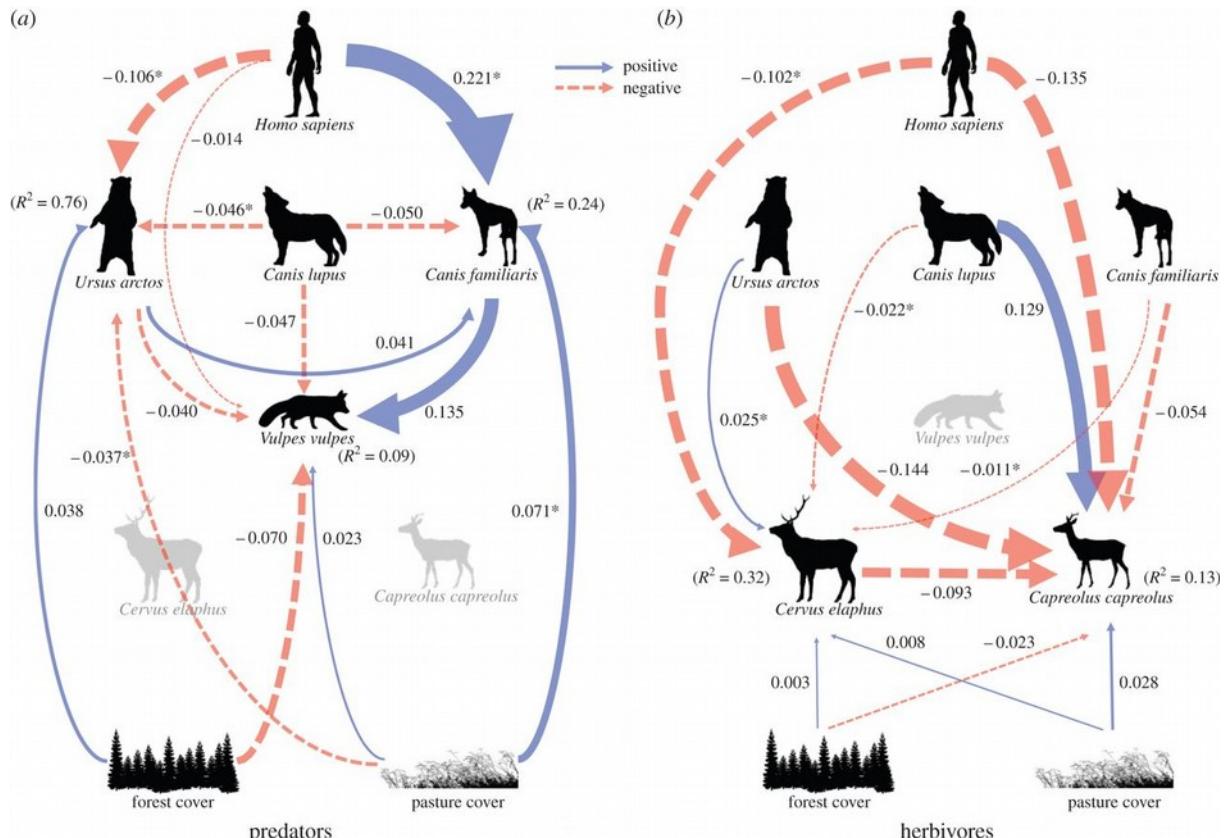


Trophic Networks

Chaînes alimentaires, réseaux écologiques



→ Principe du projet

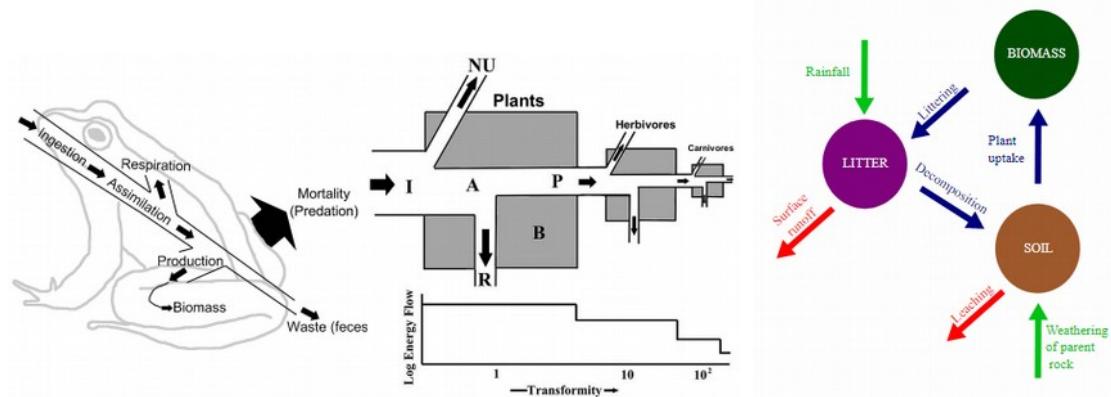
Le projet qui vous est demandé est inspiré non pas d'un jeu mais de recherches sur la stabilité des réseaux d'interactions entre espèces vivantes qui constituent les réseaux écologiques (écosystèmes).

Bien qu'il s'agisse de recherches très sérieuses visant à mieux comprendre les fondamentaux de la biodiversité et de la bonne santé des écosystèmes, nous nous proposerons ici de donner une version simplifiée et « gamifiée » permettant à un utilisateur d'explorer la structure de quelques réseaux écologiques et de manipuler interactivement les paramètres pour en mesurer les conséquences. On sera dans une démarche de sensibilisation à la complexité des écosystème et un apprentissage des **régulations** et **dynamiques** associées.

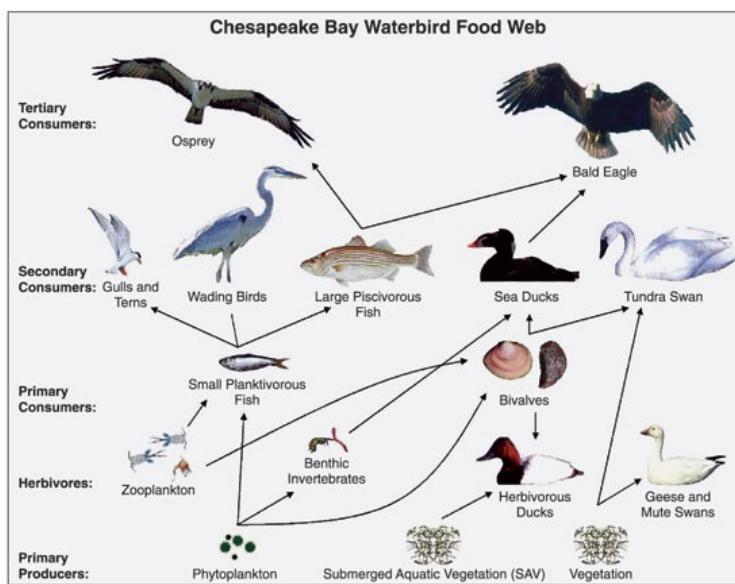
Nous mettrons en œuvre des modèles de graphes représentant des **espèces** (animales, végétales) ou **réservoirs de ressources** (lumière, sol, eau, pâture, forêt) au niveau des **sommets** et les interactions entre ces ressources et **populations** (en terme d'**échanges de biomasse** ou d'**influences**) au niveau des **arcs** (arêtes orientées). Sur cette base l'application demandée pourra permettre d'étudier interactivement des **aspects structurels** (connexité, forte connexité, k-connexité) et des **aspects fonctionnels** (dynamique des populations, dynamique des ressources).

→ Quelques exemples

On appelle réseau trophique ou plus communément « chaîne alimentaire » le réseau d'échange de biomasse constitué par les relations de consommation entre espèces (qui mange quoi/qui) avec à la base les producteurs primaires (en général les organismes utilisant la photosynthèse : les « plantes ») consommés par les animaux herbivores eux mêmes consommés par les prédateurs carnivores... A chaque niveau trophique une certaine quantité de ressources est « perdue » et rejoint la base du cycle des nutriments en passant par les détritivores (vers, limaces) et décomposeurs (bactéries, champignons saprophytes de la litière forestière...) et est en partie recyclée.



A gauche, à chaque niveau trophique des ressources sont « perdues », seule une fraction de la biomasse primaire produite se retrouve au sommet de la chaîne alimentaire. A droite, une partie des ces ressources perdues sont recyclées, une partie sort du système (et entre dans un autre, par exemple dans les effluents des fleuves) et enfin une partie se renouvelle à partir du socle minéral.

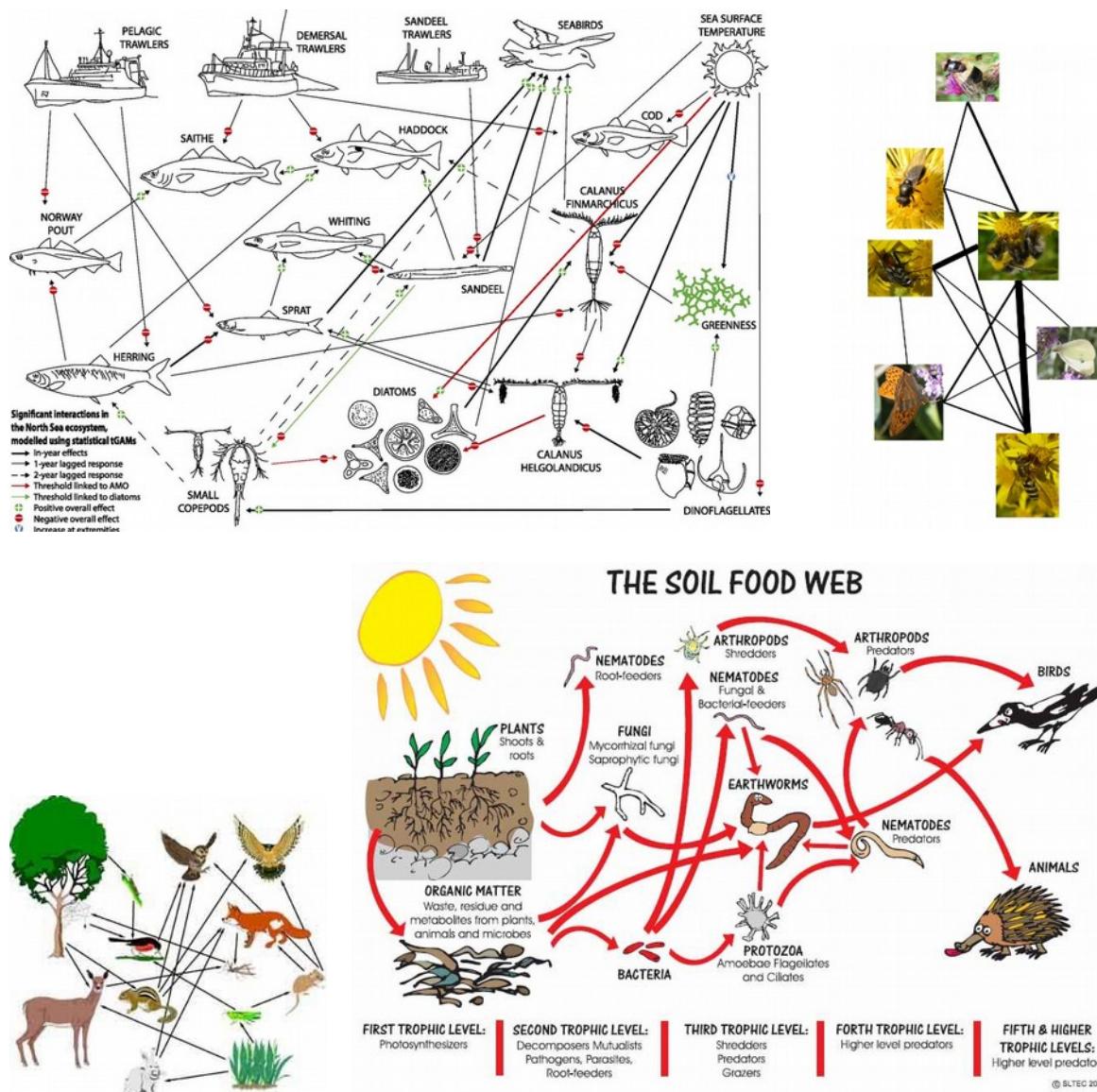


Un exemple de réseau trophique limité à un écosystème précis. **C'est ce genre de réseau qu'on souhaite modéliser dans notre application.** On pourra y ajouter des arcs descendants correspondant au cycle des déchets, par exemple dans un environnement terrestre sans apport extérieur, les nitrates des déjections animales sont utiles au renouvellement du couvert végétal. On obtiendra alors des effets de feed-back du fait de la présence de circuits.

Au delà du simple transfert de biomasse de la relation directe de consommation il existe des modèles plus fins de fonctionnements des écosystèmes mettant en jeu des relations plus complexes de type symbiose (champignons mycorhiziens associés aux arbres par exemple, essentiels aux forêts) ou parasitismes (les coucous qui utilisent des parents d'autres espèces d'oiseaux) ou compétition pour un habitat (utilisation des mêmes grottes) ou simple antagonisme (une espèce en dérange une autre, par exemple des homo-sapiens qui pique-niquent dans une réserve naturelle) : [réseau non trophique](#).

Un exemple de réseau non trophique est donné en page de garde : les arcs représentent des influençant négatives ou positives, pas nécessairement des consommations alimentaires directe.

En faisant des recherches (préférentiellement en anglais compte tenu du plus grand volume de publications scientifiques anglophones) sur tous ces mots clés, « food web » « food chain» « trophic network » « ecological network » « trophic network model » etc... vous trouverez de nombreux exemples plus ou moins complexes qui vous permettront de proposer à l'utilisateur de votre application une **base de donnée de réseaux** à étudier (au moins 3).



Quelques exemples de réseaux d'écosystèmes. Ils devront être adaptés pour « entrer » dans l'appli...

→ Cahier des charges de départ (fondations) : 6 points

Sur la base du code fourni (bibliothèque de widget graph_lib) basé sur Allegro 4, non obligatoire, ou en mettant en œuvre votre propre solution **graphique** (Allegro 4, éventuellement SDL) ou en utilisant une bibliothèque GUI C++ de haut niveau (wxWidgets, Qt... Attention ces bibliothèques sont bien documentées et solides mais nécessitent un bon niveau et un temps d'installation/apprentissage) vous proposerez **une application avec interface graphique interactive (GUI, Graphic User Interface)**.

Compte tenu de l'aspect incomplet (parfois rudimentaire) du code de base fourni, il est tout à fait acceptable d'utiliser la console en complément de l'interface graphique (par exemple pour faire des saisies de noms de fichiers images etc...). Le code de base fourni propose une implémentation de classes Vertex/Edge/Graph qu'il est possible, non obligatoire, d'utiliser. Il est possible d'utiliser la bibliothèque de widget (namespace grman) fournie sans utiliser la classe Graph fournie (càd. faire votre propre modèle de graphe). Si vous utilisez la classe Graph fournie il est normal (il faudra) la compléter, la modifier, pour l'ajuster aux besoins du projet : les classes fournies sont fonctionnelles mais incomplète, et encore en rodage.

L'application devra permettre à l'utilisateur d'**éditer** des réseaux (graphes orientés) où les noeuds représentent des espèces (avec des images déjà présentent dans le répertoire de projet, sous forme de vignettes pré-découpées) et les arcs des interactions. Ces graphes auront une mise en page (*layout de graphe*) stable, l'utilisateur pourra ajuster manuellement la position des sommets pour améliorer la lisibilité des graphes (drag & drop). Il n'est pas demandé dans le CDC de base d'automatiser la présentation (positionnement automatique des sommets : difficile).

L'application devra permettre de **charger** et de **sauver** les graphes. D'une exécution à l'autre on devra pouvoir retrouver la même structure (mêmes arcs entre mêmes sommets) et la même présentation (sommets aux même endroits, même vignettes images).

Les opérations (sauver/charger/ajouter/enlever...) se feront à partir de boutons/cases à cocher au niveau d'une barre d'outils ou directement sur les éléments (par exemple un bouton « delete » sur chaque sommet permet de le supprimer, ou bien une case select permet de sélectionner un ou plusieurs sommets et un bouton supprimer dans la barre d'outils supprime les sélectionnés).

Votre application sera livrée avec **3 réseaux écologiques** (au sens large, réseau trophique ou avec aspects non-trophiques) prêts à être utilisés (**chargés depuis 3 fichiers correspondants**). L'utilisateur pourra passer de l'un à l'autre (sauver l'actuel, charger un nouveau...) **sans quitter l'application** (et évidemment sans recompiler!). Les 3 réseaux auront été construits par vos soins sur la base de réseaux écologiques réels, éventuellement simplifiés, avec 3 niveaux de complexité croissante. Le plus complexe devra comporter au moins 12 sommets et environ une 20^{aine} d'arcs ou plus.

Remarque : compte tenu de la résolution des projecteurs il est recommandé de s'en tenir au 1024x768 comme résolution maximale, calculez dès le départ la taille de vos éléments d'interface pour que l'écran reste lisible. Cependant vous pouvez faire des essais, la résolution (livrée en 1024x768) est modifiable dans grman/grman.cpp lignes 25 et 26, testez sur projecteur école...

→ Etude des aspects structurels : 5 points

Les réseaux trophiques ou non-trophiques d'écosystèmes locaux sont généralement connexes. En revanche on peut se poser le problème de leur forte connexité. Prévoyez au moins un exemple de réseau non trivial en terme de forte connexité (càd. avec des circuits) et permettez à votre application de faire une étude de forte connexité : à l'aide d'un algorithme de forte connexité (dont on n'exigera pas que son implémentation soit « performante » : vous pouvez utiliser un algorithme « naïf »). Le résultat sera un affichage du **nombre de composante fortement connexes**, une mise en valeur visuelle de **ces composantes** (par exemple en affectant une couleur différente à chacune, ou de tout autre façon intuitive), et l'affichage graphique, temporaire, d'un **graphe réduit** du graphe de départ (*voir page 16 du poly de cours*) avec des sommets simplifiés (sans vignettes...)

Un aspect plus avancé que celui de connexité est celui de k-connexité. C'est l'étude du nombre minimum d'arcs/sommets qu'il faut enlever pour rendre le graphe non (simplement)connexe. Voir les définitions formelles >> [ici](#) et [là](#) <<. Il ne sera pas nécessaire d'enlever réellement les arcs/sommets du graphe de l'appli pour déterminer ces propriétés, il suffira par exemple d'associer à chaque arc/sommet un attribut booléen « actif » et de considérer que les éléments inactifs ne font (temporairement) plus partie du graphe.

Une espèce peut disparaître mais en revanche une relation entre espèce est stable : les chats sauvages peuvent disparaître (c'est malheureusement le cas dans une bonne partie des forêts d'Europe) mais la relation « chat sauvage » ↔ « souris » reste toujours valable (tant que les 2 sont présents). On s'intéressera donc uniquement à la k-sommet-connexité : **combien au minimum de sommets doit on enlever pour déconnecter le graphe (le rendre non simplement-connexe) et quels sont ces sommets**. L'algorithme utilisé pourra être naïf (non performant) par exemple :

Pour chaque k de 0 à ordre et tant que kmin non trouvé

Pour chaque k-plet de sommets

Si le fait de neutraliser ces k sommet déconnecte le graphe alors

kmin trouvé : kmin = k (mais on termine bien la boucle « Pour chaque k-plet... »)

Enregistrer ce k-plet de sommets (pour montrer après)

$$\binom{n}{k} = C_n^k = \frac{n!}{k!(n-k)!}$$

avec n = ordre

Supposons qu'aucune façon d'enlever 1 puis 2 sommets ne déconnecte le graphe, mais on trouve 5 triplets de sommets qui, quand on neutralise le triplet, déconnectent le graphe. Alors on indiquera que le graphe est 3-connexe (au sens de la k-sommet-connexité) et on affichera, si possible de façon visuelle (en faisant défiler?) les 5 triplets en montrant à chaque fois les composantes connexes séparées par ces déconnexions. Le nombre de k-plets dépend du coefficient binomial... Cette valeur peut devenir très grande du fait des factorielles : prévoyez des limites.

Du point de vue de l'étude des écosystème, plus le k de la k-sommet-connexité est élevé plus on pourra considérer (de façon simplifiée) que le système est robuste : la disparition de plusieurs espèce ne rendrait pas le système déconnecté (avec disparitions en chaîne qui s'ensuivent). En réalité ceci dépend de l'**importance relative** des relations. Ainsi même si les chats sauvages mangent régulièrement des sauterelles, leur aliment principal reste les souris. La disparition (peu probable) des souris ne déconnecterait pas (au sens topologique) les chats sauvages du producteur primaire (l'herbe, mangée par les souris et les sauterelles) mais remettrait certainement en cause leur survie : un chat sauvage ne peut pas manger que des sauterelles. C'est l'objet des études fonctionnelles.

→ Etude des aspects fonctionnels : 5 points

On va donc s'intéresser à des aspects **quantitatifs** et **dynamiques** : comment est-ce que le réseau fonctionne en terme de flux, quelle est l'**évolution temporelle** des paramètres en fonctions des coefficients d'influences et d'échanges. Le plus évident, sur la base d'un réseau trophique, est de s'intéresser à la **dynamique des populations**.

On supposera que la « santé » d'une espèce se limite à la taille de sa population. Chaque sommet du graphe aura donc une valeur N associée. On pourrait considérer que N est le nombre d'individus (plutôt pour des loups), ou le nombre de kilos (la biomasse) d'individus représentant l'espèce (plutôt pour des brins d'herbes). A notre niveau il ne sera pas nécessaire de calibrer ces valeurs de N sur des unités précises, on pourra arbitrairement décider que N=100 représente une population de taille optimale (stable par rapport aux autres).

Une population a une taille qui croît proportionnellement d'autant plus vite qu'il y a de représentants : 1 couple de lapins peut « fabriquer » 10 nouveaux lapins en 3 mois, 10 couples de lapins peuvent « fabriquer » 100 nouveaux lapins en 3 mois. C'est le principe de l'équation du modèle exponentiel de croissance. Mais ceci ne tient pas compte des ressources (donc des sommets prédeesseurs dans le réseau trophique) ni des prédatations (sommets successeurs dans le réseau trophique). On adoptera donc un modèle logistique de dynamique de population :

$N_{t+1} = N_t + rN_t(1 - N_t/K)$ avec **population à l'instant t+1** (un tour de boucle de jeu de notre projet) en fonction de la **population à l'instant t**, **r rythme de croissance** (valeur à diminuer si le système est instable et les populations évoluent trop rapidement) et **K est la capacité de portage de l'environnement**.

Par exemple on peut imaginer que sur une prairie de 100m² il y a 100 kg d'herbe fraîche produite par mois, ce qui permet de faire vivre K=100 lapins. Si on met N=50 lapins dans cet écosystème, alors N_t/K<1 et donc N_{t+1}>N_t (la population grossit). À l'inverse si on débarque N=200 lapins, alors N_t/K>1 et donc N_{t+1}<N_t (la population diminue). Ceci constitue donc une **régulation** (homéostasie). La régulation est stable (N converge vers K) si r est « assez petit » (à régler empiriquement).

Là où les couplages (les arcs entre sommets) interviennent c'est dans la détermination du K : en effet si la prairie ne produit plus que 50 kilos d'herbe fraîche par mois alors le K des lapins est divisé par 2. On pourra donc associer à chaque arc un coefficient qui représentera la contribution de la population du sommet de départ (le mangé) à la valeur de K du sommet d'arrivée (le mangeur). Ces contributions se sommeront : si les lapins mangent de l'herbe et des carottes sauvages, $K_{\text{lapin}} = \text{Coef}_{\text{herbe} \rightarrow \text{lapin}} \times N_{\text{herbe}} + \text{Coef}_{\text{carottes} \rightarrow \text{lapin}} \times N_{\text{carottes}} + \dots$ pour tous les arcs prédeesseurs .

En retour, les quantités consommées (prédatation) viennent se **soustraire** directement à la population :

$$N_{\text{herbe}}(t+1) = N_{\text{herbe}} + r_{\text{herbe}} N_{\text{herbe}} (1 - N_{\text{herbe}} / K_{\text{herbe}}) - \text{Coef}_{\text{herbe} \rightarrow \text{lapin}} \times N_{\text{lapin}} - \text{Coef}_{\text{herbe} \rightarrow \text{chevreuil}} \times N_{\text{chevreuil}} \dots$$

De la même façon le lapin se verra retranché à sa formule la somme pondérée des prélèvements des renards et loups et lynx etc... Ceci établit donc une rétroaction (feedback) même si le graphe ne comporte pas de circuit, l'arc bien qu'orienté du consommé vers le consommateur « agit dans les 2 sens » puisque le consommé est modifié. Ainsi la population N du loup modifie au final la population N d'herbe (en limitant les populations de lapins qui consomment l'herbe).

Remarque : il sera peut-être nécessaire d'introduire un coefficient pour pondérer la valeur de ce terme négatif (prédatation) car le modèle proposé n'est pas étalonné sur des échelles homogènes (par exemple où chaque N serait évalué en kg de biomasse).

N'hésitez pas à expérimenter pour trouver des réglage « intéressants » des coefficients, c'est justement l'intérêt de cette application d'autoriser des approches ludiques/empiriques d'exploration des dynamique. Une dynamique « intéressante » est une dynamique stable qui ne diverge pas (en + ou en - n'oubliez pas que si un calcul donne $N < 0$ alors $N = 0$: il n'y a pas de population négative!) et qui présente des oscillations.

Sans avoir à justifier quantitativement les valeurs des paramètres votre application montrera « **en temps réel** » **les valeurs des population évoluer selon les formules du modèle logistique**. Elle permettra de lancer/pauser l'animation des valeurs, de modifier les valeurs (des paramètres et/ou des population) et de reprendre la simulation pour voir les conséquence de l'intervention. On pourra aussi voir la conséquence de la neutralisation des sommets identifiés lors de l'analyse structurale de k-connexité : que se passe-t-il si les espèces qui garantissent la connexité de l'écosystème disparaissent ? Pour « neutraliser » un sommet il suffira par exemple de mettre sa population à 0 !

Remarque technique : un modèle « discret » $N_{t+1} = f(N_t)$ est en général une approximation d'un modèle en temps continue (équation différentielle). En utilisant des coefficients r assez petits le résultat de la simulation en temps discrets (tours de boucle) s'approche asymptotiquement du résultat qu'on obtiendrait avec le modèle continu. Il s'agit d'une intégration numérique du 1^{er} ordre... En principe, informatiquement, les mises à jour des valeurs dynamiques (Les N_t des différents sommets) devraient se faire de façon synchrone, ce qui impliquerait de bufférer les nouvelles valeurs avant de les actualiser. En pratique on pourra se contenter de faire le calcul $N_{t+1} = f(N_t)$ à tour de rôle pour chaque élément. Il s'agit d'une approche simplifiée et approximative de la simulation.

Les coefficients permettant d'obtenir des dynamiques « intéressantes » devront pouvoir être sauvé/restitués dans le fichier de sauvegarde associé au graphe, avec les populations à un moment donné.

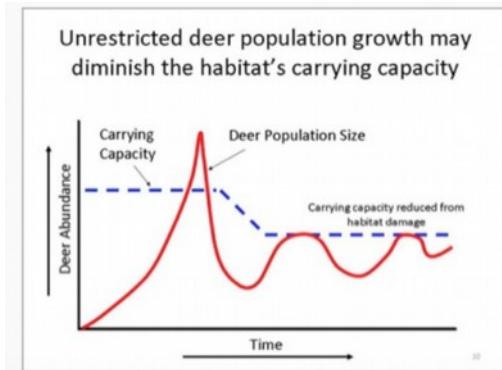
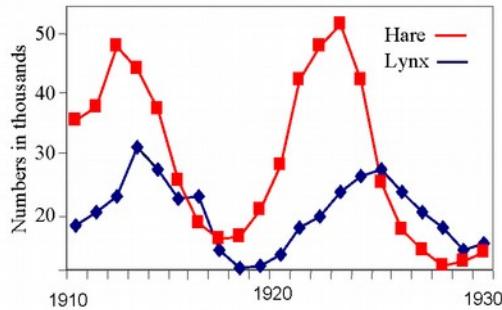
→ Extensions : 4 points

Une fois les études des aspects structurels et fonctionnels intégré(s) à votre projet vous pouvez envisager les extensions suivantes. Le jury évaluera chacun de ces points d'extension en fonction de la difficulté technique, du volume de travail de programmation, de la qualité du résultat et de la conception du code. Il n'est pas demandé de cumuler toutes les extensions. Le nombre de points d'extensions est limité, il n'est pas possible de cumuler plus de 4 points d'extension pour « compenser » des faiblesses par ailleurs.

- **Habillement : améliorations de présentation, d'interface...**
 - Compléter les widgets de la bibliothèque proposée, boites de saisie clavier...
 - Représentations améliorées : l'épaisseur des arcs représente leur importance etc ...
 - Proposer des animations interactives : des flots de petits lapins défilent vers le renard...

- **Moteur de simulation**

- Vous n'aimez pas le coté empirique des réglages suggérés pour l'étude fonctionnelle : faites valoir la toute puissance du système métrique en homogénéisant les biomasses en kg, les flux énergétiques en watts, et en adoptant des coefficients réaliste issus de la littérature (1kg de bœuf pour 7kg de céréales ...), en mesurant le CO₂ absorbé/émis ...
- Votre ambition mathématique dévorante vous pousse à affiner l'intégration numérique, voyez les méthodes du 2ème ordre, multi-pas etc...
- Intégrez des aspects anthropiques (facteurs humains) comme l'impact de la dégradation des habitats sur les population (coefficients modifiés, capacités de portage K bornées du fait de la destruction de sites de nidification...)
- Complétez le modèle de dynamique de population en intégrant des contraintes multifactorielles, par exemple en suivant quantitativement les flux de plusieurs nutriments. Voir par exemple comment [les arbres ont besoin de saumons](#).
- Représentez l'évolution temporelle des populations pas juste avec des valeurs qui bougent mais avec enregistrements et présentation dans des graphiques (*charts*)



- **Recherche : écumez les publications scientifiques sur le sujet**

- La motivation pour ce sujet est partie d'un article fascinant : [Trophic coherence determines food-web stability](#)
 Sans entrer dans tous les détails des valeurs propres des matrices aléatoire, sauriez vous survoler cet article, en extraire les idées générales, et proposer une démonstration crédible de l'idée selon laquelle la « cohérence trophique » (aspect structurel mesurable sur vos graphes) impacte bien positivement la corrélation entre biodiversité et stabilité ?
- Il existe pléthore d'autres publications sur le sujet, trouvez votre article et essayez de reproduire une [mesure de propriété](#) (connectance / clustering / nestedness ...) et démontrez une corrélation sur un cas concret.
- Développez l'aspect recyclage en modélisant les liens descendants (détritivores)
- Ouvrez des horizons, il n'y a pas que manger et être mangé dans la vie, il y a aussi des symbioses, des [réseaux de pollinisation](#) ...

- Autres extensions

- ... soyez créatifs !

→ Consignes générales :

- Il est autorisé d'utiliser des graphismes tiers trouvés sur Internet, si possible libres de droits (*Creative commons...*) ou avec *Copyright*. Si possible les sources tierces doivent être citées dans un fichier **licence.txt** dans votre archive.
- Il est autorisé d'utiliser/adapter les codes sources publiés sur [le site du cours allegro](#), ainsi que des **extraits** de code source libre de droit (*public domain*, GPL, LGPL ...) en précisant ces emprunts. Les commentaires initiaux de ces codes sources empruntés doivent être mis à jour par vos soins pour refléter leur utilisation dans votre contexte (appropriation correcte des codes tiers utilisés). En soutenance vous devez être capable d'expliquer chaque ligne de code de votre projet déposé. De façon générale il est possible d'utiliser des bibliothèques du domaine public.

En dehors de ces cas, toute détection de copie massive de code tiers sera considéré comme plagiat et très sévèrement sanctionné, avec circonstances aggravantes si

- l'emprunt n'est pas cité
- l'emprunt vient du travail d'une autre équipe d'étudiants de l'ECE pour ce même projet
- il y a tentative de dissimulation (maquillage de code...)

- Le code source doit être présentable : bien indenté, raisonnablement aéré et commenté.

→ Consignes d'organisation

Les consignes de dépôt, dates de rendu, conditions, équipes de projets, sont données sur campus.
Le présent document constitue le cahiers des charges technique du projet.

Robin Fercoq et toute l'équipe encadrante,
02/04/2018

Version 1, sujette à ajustements mineurs en cas de besoin.

Toutes les images citées dans ce sujet sont Copyright de leurs créateurs respectifs

