**Trophic Networks**

Cahier des charges

Nous métrons en œuvre des modèles de graphes représentant des espèce (animales, végétales) ou réservoir de ressources (lumière, sol, eau, pâture, forêt) au niveau des sommet et les interactions entre ces ressources et populations (en termes 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 aspect fonctionnel (dynamique des populations, dynamique des ressources).

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.

- une base de données au moins 3

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 rénaux à étudier (au moins 3).

Cahier des charges de départ (fondations) : 6 points

-Une application avec interface graphique interactive

Possible d’utiliser le code de fercoq.

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….

-l’application devra permettre à l’utilisateur d’éditer des réseaux où les nœuds représentent des espèces (avec des images déjà présentent dans le répertoire de projet, sous forme de vignettes prédécoupées) et les arcs des interactions : L'utilisateur pourra ajuster manuellement la position des sommets pour améliorer la lisibilité des graphes (drag & drop).

Ces graphes auront une mise en page (layout de graphe) stable.

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 mêmes 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 selecte 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 fichier correspondant). 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 20aine d'arcs ou plus.

Etude des aspects structurels : 5 points

Connexité :

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 termes 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...)

Bonus : 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. 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 parte du graphe.

Une espèce peut disparaître mais sa relation entre une autre espèce reste toujours valable tant qu’elle n’a pas encore disparu : à revoir la partie de k connexité avec l’algorithme.

Etude des aspects fonctionnels :

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