Дніпровський національний університет імЕНІ Олеся Гончара

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ МАТЕМАТИКИ ТА

МАТЕМАТИЧНОЇ КІБЕРНЕТИКИ

**КУРСОВА РОБОТА**

за спеціальністю

на тему **«Розробка математичної моделі групової взаємодії бджіл»**

Виконав:

студент групи ПА-17м-1

спеціальності 113«Прикладна математика»

Кривоносов Олександр Дмитрович

Керівник: доцент кафедри ОМ та МК

Кузенков Олександр Олександрович

Кількість балів\_\_\_\_\_\_\_\_\_\_\_\_

Національна шкала \_\_\_\_\_\_\_\_

Оцінка ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії:

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

м. Дніпро

2018

ЗМІСТ

ВСТУП 3

1. ПОСТАНОВКА ЗАДАЧІ 5

2. ОГЛЯД ПРОБЛЕМИ 7

3. ОСНОВНА ЧАСТИНА 9

3.1 Структура моделі 9

3.2 Модель колонії 10

3.3 Модель кліща 11

3.3 Модель кормів 12

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 15

ДОДАТОК А 16

### ВСТУП

Існує клас об'єктів, для яких з різних причин не розроблені аналітичні моделі, або розроблені моделі не підходять для проведення результативних обчислювальних експериментів. У цьому випадку математична модель замінюється імітаційної моделлю.

Імітаційне моделювання­ – це окремий випадок математичного моделювання, завдяки якому можна побудувати модель, що описує процес так, як він відбувався б у дійсності. Таку модель можна «програти» в часі як для одного випробування, так і декількох, при цьому результати будуть визначатися випадковим характером процесів.

У імітаційному моделюванні можна виділити наступні основні підходи: динамічні системи, системна динаміка, дискретно-подійове моделювання, мультиагентне моделювання.

Мультиагентна система - це система, утворена декількома взаємодіючими інтелектуальними агентами. Мультиагентні системи можуть бути використані для вирішення таких проблем, які складно або неможливо вирішити за допомогою одного агента або монолітної системи. Для розробки мультиагентної системами використовують Мультиагентні моделювання.

На відміну від системної динаміки і дискретно-подієвих моделей мультиагентні моделі децентралізовані. В процесі побудови такої моделі не визначається поведінка системи в цілому, визначається поведінку агентів на індивідуальному рівні, а динаміка системи виникає як результат діяльності багатьох агентів. Мультиагентне моделювання є підходом більш універсальним і потужним, оскільки воно дозволяє врахувати будь-які складні структури і поведінки. Інша важлива перевага мультиагентного моделювання в тому, що розробка моделі можлива без знання про глобальні залежності: потрібно визначати індивідуальну логіку поведінки учасників процесу для того, щоб побудувати мультиагентну модель і вивести з неї глобальну поведінку. Мультиагентну модель простіше підтримувати: уточнення зазвичай робляться на локальному рівні і не вимагають глобальних змін. Мультіагентну модель реалізують клітковим автомат.

Бджолина колонія є мультиагентною системаою, тому для її моделювання підходить мультиагентній підхід. Модель бджолиної колонії можна використовувати для короткочасних і довготривалих прогнозів поведінки конкретної колонії бджіл. Для цього потрібно періодично корегувати параметри моделі щоб вони співпадали з вже відомими параметрами колонії. Щоб зробити процес корегування швидким і зручним можна створити систему для зберігання і аналізу даних.

### 1. ПОСТАНОВКА ЗАДАЧІ

***Означення 1***

Вулик - штучне житло, виготовлене людиною для утримання медоносних бджіл. Залежно від системи в одному вулику може жити одна або кілька сімей бджіл. Способи утримання бджіл у вуликах різних систем також не однакові.

***Означення 2***

Бджолина сім'я — колонія, що складається з декількох тисяч робочих бджіл, однієї плодовитої матки і декількох сотень (іноді тисяч) трутнів.

***Означення 3***

Матка — це єдина повністю розвинена жіноча особина в сім'ї. Матку бджолярі часто називають «королевою». Матка може відкладати два види яєць: запліднені та незапліднені. Це залежить від того, в яку комірку матка опустить своє брюшко. Із запліднених яєць розвиваються робочі бджоли або матки (в залежності від типу годівлі личинок), а з незапліднених - трутні.

***Означення 4***

Робочі бджоли — це недорозвинені жіночі особини. Тобто це фактично жіночі особини, але з недорозвиненими статевими органами. Виконують практично всі роботи із життєзабезпечення сім'ї: прибирання вулика, чистку осередків, будівництво стільників, охорону вулика, збір нектару, пилку, прополісу, вигодовування личинок.

***Означення 5***

Трутні — це самці, які входять до складу бджолиної сім'ї з початку весни до осені. У трутня є єдине дане йому природою призначення - це запліднення маток.

***Означення 6***

Розплід — бджолині яйця, личинки, лялечки, що розвиваються у сотах.

***Постановка задачі.***

Створити математичну модель бжолиної колонії. Для цього необхідно вирішити наступні завдання:

• зібрати, проаналізувати і структурувати інформацію про бджолину колонію;

• вибрати середу моделювання для реалізації моделі;

• реалізувати базову модель, внутрішній стан колнії;

• на основі базової моделі створити моделі більш високої деталізації.

### 2. ОГЛЯД ПРОБЛЕМИ

На бджолину колонію впливають різноманітні фактори, у тому числі кліщі варроа, інфекції, зміни ландшафтної структури, пестициди та інше. Для автоматизації висновків і прогнозу стану колонії може бути корисним створення програми, яка допоможе пасічникам доглядати пасіку. Основою такої програми має бути модель бджолиної колонії, тому і виникає потреба в ній.

Існує клас об'єктів, для яких з різних причин не розроблені аналітичні моделі, або розроблені моделі не підходять для проведення результативних обчислювальних експериментів. У цьому випадку математична модель замінюється імітаційної моделлю.

Імітаційне моделювання­ – це окремий випадок математичного моделювання, завдяки якому можна побудувати модель, що описує процес так, як він відбувався б у дійсності. Таку модель можна «програти» в часі як для одного випробування, так і декількох, при цьому результати будуть визначатися випадковим характером процесів.

У імітаційному моделюванні можна виділити наступні основні підходи: динамічні системи, системна динаміка, дискретно-подійове моделювання, мультиагентне моделювання.

Мультиагентна система – це система, утворена декількома взаємодіючими інтелектуальними агентами. Мультиагентні системи можуть бути використані для вирішення таких проблем, які складно або неможливо вирішити за допомогою одного агента або монолітної системи. Для розробки мультиагентної системами використовують Мультиагентні моделювання.

На відміну від системної динаміки і дискретно-подієвих моделей мультиагентні моделі децентралізовані. В процесі побудови такої моделі не визначається поведінка системи в цілому, визначається поведінку агентів на індивідуальному рівні, а динаміка системи виникає як результат діяльності багатьох агентів. Мультиагентне моделювання є підходом більш універсальним і потужним, оскільки воно дозволяє врахувати будь-які складні структури і поведінки. Інша важлива перевага мультиагентного моделювання в тому, що розробка моделі можлива без знання про глобальні залежності: потрібно визначати індивідуальну логіку поведінки учасників процесу для того, щоб побудувати мультиагентну модель і вивести з неї глобальну поведінку. Мультиагентну модель простіше підтримувати: уточнення зазвичай робляться на локальному рівні і не вимагають глобальних змін. Мультіагентну модель реалізують клітковим автомат.

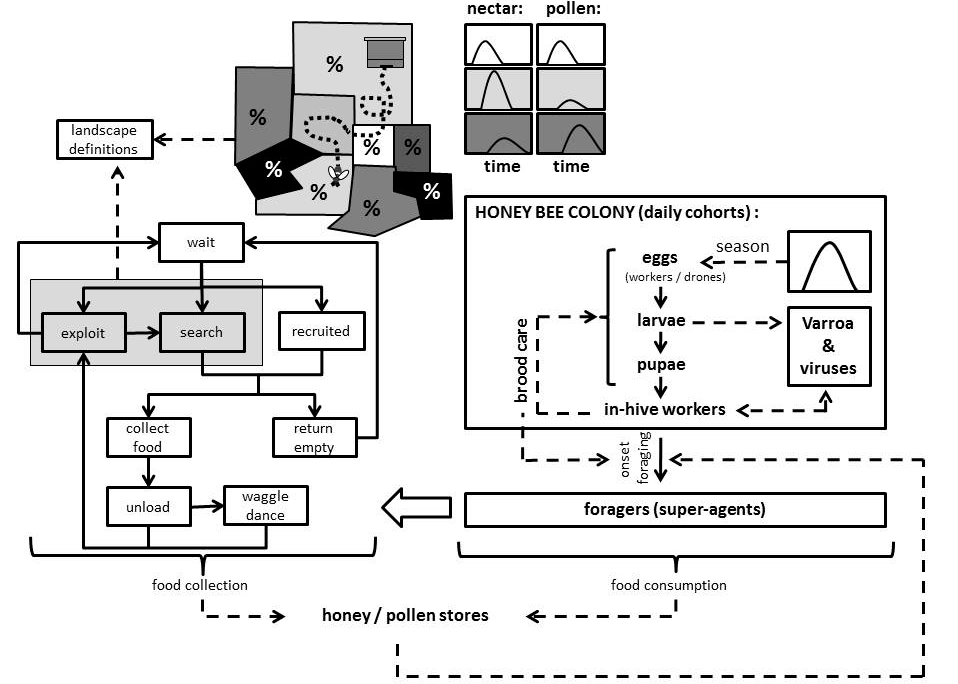
Бджолина колонія є мультиагентною системаою, тому для її моделювання підходить мультиагентній підхід. Модель бджолиної колонії можна використовувати для короткочасних і довготривалих прогнозів поведінки конкретної колонії бджіл. Для цього потрібно періодично корегувати параметри моделі щоб вони співпадали з вже відомими параметрами колонії. Щоб зробити процес корегування швидким і зручним можна створити систему для зберігання і аналізу даних.

### 3. ОСНОВНА ЧАСТИНА

### 3.1 Структура моделі

Опис моделі відповідає ODD [2][7] (Overview, Design concepts, Details) протоколу. Такий опис моделі облегшує розуміння моделей на основі агентної системи.

Модель складається з трьох пов'язаних моделей: моделі колонії, моделі кліща, і кормової моделі (рис.1). Модель колонії описує процеси, що входять до складу колонії. Вона базується на різнорідних рівняннях, які створюють структуру колонії та динаміку популяції колоній. Модель кліща описує динаміку популяції кліща варроа в колонії медоносних бджіл. В якості рознощиків вірусів кліщі впливають на смертність бджолиних лялечок та дорослих бджіл.

Рис.1 - Спрощений вид структури моделі

Кормова модель - індивідуальна модель, яка представляє собою здобич з квіткових полів, розташованих навколо вулика. Властивості цих квіткових полів: ймовірність виявлення робочими бджолами, відстань до вуликів або наявність нектару та пилку. Динаміка ландшафту, в тому числі динамічне розташування та наявність культурних полів різних типів, може бути врахована шляхом оновлення імпортованих даних про ландшафт кожного етапу моделі колонії.

Структура моделі являє собою компроміс між структурним реалізмом з одного боку, тобто, здатність представляти неоднорідність, а також обчислювальна ефективність щодо параметризації та аналізу моделей, з іншого боку. Концептуально, модель дуже проста і включає багато спрощених припущень. Тим не менше, з точки зору кількості об'єктів, змінних стану, параметрів і процесів, модель є досить складною.

### 3.2 Модель колонії

Сутності, що входять у колонію: групи яєць, личинок, лялечок, долослих бджіл, як робочих бджіл, так і трутнів, збережені у вулику пилок і мед, королева. Бджоли у колонії розбиті на групи, такі гркпи описуються ідентифікаційним номером, віком, статтю, кількістю бджіл у групі, кількістю мертвіх, впливом на групу кліщів та вірусних інфекцій, а також допоміжними змінними, які відстежують смертність та інфекції. Отже, кожна сім'я характеризується своїм унікальним ідентифікаційним номером, віком, числом осіб, та кількістю осіб, що загинули протягом певного періоду через смертність, та вторгнення кліщів. Сім'ї робітників та лялечок додатково характеризуються числом "infectedAsPupa" та "number\_healthy", що представляють кількість здорових бджіл та бджіл інфікованих. У колонії також можуть заразитися дорослі (number\_infectedAsAdult). Молоді бджоли, які ще не вміють літати, доглядають розплід (виводок), а бджоли, що збирають їжу, також беруть участь у розвитку колонії так як вони частину свого часу турбуються про розплід. Це необхідно для початку догляду за розплодом у весняний період часу, коли немає молодих бджіл, а є лише бджоли, що збирають їжу та розплід.

Вулик характеризуються значеннями "Max\_Honey\_Store\_kg" і "Max\_Broodcells", які показують максимальну кількість меду у кілограмах, що може зберігатися у вулику і максимальну кількість місця для розплоду відповідно.

Королева визначається швидкістю відкладання яєць, на що вплівають вік королеви і кількість робочіх бджіл вільних для догляду за розплодом.

### 3.3 Модель кліща

Модель кліща описує динаміку популяції кліща варроа в межах бджолиної колонії. Як рознощики вірусів (вірус деформованого крила, DWV чи вірус гострого паралічу, APV), кліщі можуть вплинути на смертність бджолиних лялечок та дорослих бджіл. Сутність моделі кліща є загальна чисельність кліщів, тобто кліщів, прикріплених до дорослих бджіл, які характеризуються двома значеннями: загальна кількість форетичних кліщів "PhoreticMites" та прпорція здорових (не заразних) кліщів "PhoreticMitesHealthyRate". Для розмноження кліщі потрапляють до комірки соту з личинкою робочої бджоли чи трутня незадовго до того, як комірка буде запечатана. Таким чином, в одній комірці личинки робочої бджоли може бути до 8 кліщів, а в комірці личинки трутня до 16 кліщів. Знати до скількох комірок скільки кліщів вторглися на певний день важливо, бо кількість кліщів у комірці лялечки визначає передачу вірусу від кліщів до лялечки та від зараженої лялечки до кліщів.

Проте в моделі колонії бджоли одного віку та статі не відрізняються індивідуально, тому для вторгнення кліща тимчасово представлені комірки личинок індивідуально і вони перевіряються на кожного кліща. Тоді допоміжний об'єкт "miteOrganiser" використовується для відстежування всіх кліщів, які вторглися в комірки личинок на ітераційний крок. Таким чином збінні стану об’єкта "miteOrganiser" такі: списоки для комірок личинок робочих бджіл і трутнів, які визначают скільки комірок скількома кліщами було уражено; пропорція здорових кліщів на день вторгнення "invadedMitesHealthyRate", що необхідно для розрахунку передачі вірусу від кліщів до лялечки та від зараженої лялечки до кліщів.

### 3.4 Модель кормів

Модель кормів включає в себе два типи суб'єктів, бджоли годівники та квіткові поля. Бджоли годівники це насправді ескадрилья робочих бджіл, що представляють певну кількість робочих бджіл, які здобувають їжу у схожий спосіб (на одних і тих галявинах); для стислості, надалі ми будемо називати ці ескадрильї як годівники. Годівники використовуються для збереження часу обчислень. Кількість бджіл у конкретній групі годівників записується в змінну "Squadron\_Size" і значенням цієї змінної можна брати за 100. Групи годівників створюються із дорослих робочих бджіл, які досягли віку, коли вже вміють літати. Група годівників характеризується змінними стану, які вказані в таблиці (Таб.1); змінна "activity" описує поточній тип одного з семи типів активності групи (Тал.2).

Квіткові поля представлені об’єктами, що надають нектар та пилок, наприклад, посівні поля або сади. Оскільки кормова модель є просторово неявною, квіткові поля не мають просторової міри або форми, але характеризуються змінними стану, що перелічені в (Таб.3).

|  |  |
| --- | --- |
| age | Вік (днів) |
| activity | Активність (Таб.2) |
| activityList | Запис усіх дій протягом дня |
| knownNectarPatch | Ідентифікатор поля з нектаром |
| knownPollenPatch | Ідентифікатор поля з пилком |
| pollenForager | [*true*/*false*] якщо *false* – збирають нектар |
| cropEnergyLoad | Енергія, що дає бджолі нектар |
| collectedPollen | Зібраний пилок |
| infectionState | Чи інфіковано бджолу [*здорова*/ *інфікована як личинка*/*інфікована як доросла особа*] |
| mileometer | Шлях, який пролетіла бджола [km] |
| km\_today | Шлях бджоли за день [km] |

Таб.1 — Змінні стану групи годівників.

|  |  |
| --- | --- |
| lazy | Ліниві годівники не працбватимуть в цей день |
| resting | Відпочиваючі бджоли, можуть бути набрані на збірання їжі «танцюючою» бджолою або можуть спонтанно почати працювати |
| expForaging | Досвідчені годівники, що збирає нектар або пилку з певного поля |
| searching | Шукають нове квіткове поле для нектару або пилку |
| bringingNectar | Приносять нектар. |
| bringingPollen | Приносять пилок. |
| recForaging | Група бджіл, яких взяла на збирання їжі «танцююча» бджола |

Таб.2 — Типи активності групи годівників.

|  |  |
| --- | --- |
| patchType | Тип квітів |
| distanceToColony | Відстань до колонії [m] |
| quantityMyl | Кількість доступного нектару [μl] |
| nectarConcFlowerPatch | Концентрація цукру в нектарі [mol/l] |
| amountPollen\_g | Кількість доступного пилку [g] |
| detectionProbability | Імовірність того, що годівник знаходить квіткове поле |
| handlingTimeNectar handlingTimePollen | Час обробки (час, що витрачуется, щоб набрати набрати нектар або пилок), залежно від виснаження нектару або пилку в квітковому полі |
| flightCostsNectar, flightCostsPollen | Енергетичні витрати, обчислюються з відстані до колонії та часу обробки |
| eef | Енергетична ефективність, розрахована з концентрації нектару та витрат на збирання |
| danceCircuits | Показник сили танцю, підраховуется з EEF |
| danceFollowersNectar | Кількість можливих годівників, розрахованих з сили танцю |
| summedVisitors | Кількість візитів годівників |
| nectarVisitsToday, pollenVisitsToday | Кількість візитів за нектаром, пилком сьогодні |
| tripDurationNectar | Час, що витрачается на подорож за нектаром |
| tripDurationPollen | Час, що витрачается на подорож за пилком |
| mortalityRisk | Ризик вмирати під час подорожі за нектаром, розрахований з tripDurationNectar та  handlingTimeNectar |
| mortalityRiskPollen | Ризик вмирати під час подорожі за пилком, розрахований з tripDurationPollen та  handlingTimePollen |

Таб.3 — Змінні стану квіткового поля.

### СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

 1. Буренин Н.Л. Справочник по пчеловодству / Н.Л. Буренин, Г.Н. Котова. — Москва, 2005.

 2. De Vries, H. & Biesmeijer, J.C. (2002) Self-organization in collective honeybee foraging: emergence of symmetry breaking, cross inhibition and equal harvest-rate distribution. [UKR] Самоорганізація в колективному харчуванні медових бджіл: виникнення порушення симетрії, перехресного гальмування та рівномірного розподілу урожаю. Behavioral Ecology and Sociobiology, 51, 557-569.

3. Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G., Huth, A., Jepsen, J.U., Jørgensen, C., Mooij, W.M., Müller, B., Pe’er, G., Piou, C., Railsback, S.F., Robbins, A.M., Robbins, M.M., Rossmanith, E., Rüger, N., Strand, E., Souissi, S., Stillman, R.A., Vabø, R., Visser, U. & DeAngelis DL (2006) Стандартний протокол для описання моделей на основі індивідуальних характеристик та на основі агентних систем. *Ecological Modelling,* 198, 115-126.

4. Martin, S. (1998) A population model for the ectoparasitic mite Varroa jacobsoni in honey bee (Apis mellifera) colonies. Ecological Modelling, 109, 267–281.

5. Pankiw, T., Huang, Z.Y., Winston, M.L. & Robinson, G.E. (1998) Queen mandibular gland pheromone influences worker honey bee (Apis mellifera L.) foraging ontogeny and juvenile hormone titers. Journal of Insect Physiology, 44, 685-692.

6. Toth, A.L., Kantarovich, S., Meisel, A.F. & Robinson, G.E. (2005) Nutritional status influences socially regulated foraging ontogeny in honey bees. Journal of Experimental Biology, 208, 4641-4649.

7. Grimm, V., Berger, U., DeAngelis, D.L., Polhill, G., Giske, J. & Railsback, S. F. 2010 The ODD protocol: a review and first update. Ecological Modelling, 221, 2760-2768.

### ДОДАТОК А

**Код, що описує структуру сутностей моделі**

#include "impl\_bee\_model.h"

namespace My {

namespace BeeModel {

struct internal\_model\_data {

Impl::model\_impl model;

internal\_model\_data(std::shared\_ptr<model\_data> data) : model(std::move(data)) {}

};

model:model() : model(std::make\_shared<model\_data>(model\_data::create\_sample())) {}

model::model(std::shared\_ptr<model\_data> data) {

const char\* info;

if (!Impl::verify\_data(\*data, info)) {

throw std::invalid\_argument(info);

}

idata.reset(new internal\_model\_data(std::move(data)));

}

const model\_data& model::get\_data() {

return idata->model.get\_data();

}

model::daily\_step() {

idata->daily\_step();

}

} // namespace BeeModel

} // namespace My

//bee\_model.h

#ifndef \_\_BEEMODEL\_H\_\_

#define \_\_BEEMODEL\_H\_\_

#include <string>

#include <list>

#include <memory>

#include <My/Properties.h>

#define DAYS\_IN\_YEAR (365)

typedef unsigned int UINT;

typedef unsigned short USHORT;

namespace My {

namespace BeeModel {

struct date\_struct {

USHORT year = 2000;

USHORT day = 1; //from 1 to DAYS\_IN\_YEAR //Day

void inc();

};

struct entity {

int64\_t id = 0;

USHORT age = 0;

int ploidy = 0;

USHORT number = 0;

USHORT numberDied = 0;

USHORT invadedByMiteOrganiserID = 0;

};

struct egg\_group : entity {};

struct egg\_drone\_group : entity {};

struct larva\_group : entity {};

struct larva\_drone\_group : entity {};

struct base\_bee\_group : entity {

USHORT infectedAsPupa = 0;

USHORT healthy = 0;

};

struct pupa\_group : base\_bee\_group {};

struct pupa\_drone\_group : base\_bee\_group {};

struct drone\_group : base\_bee\_group {};

struct in\_hive\_bee\_group : base\_bee\_group {

USHORT infectedAsAdult = 0;

};

struct forager\_squadron : entity { //bee group, which searching food

int activity = 0;

int activityList = 0;

int knownNectarPatch = 0;

int knownPollenPatch = 0;

int pollenForager = 0;

int cropEnergyLoad = 0;

int collectedPollen = 0;

int mileometer = 0;

int km\_today = 0;

int infectionState = 0;

};

struct flower\_patch : entity {

struct daily\_data {

UINT quantityMyl = 0; // [μl] microliters; quantity of available nectar on the specified day

UINT amountPollen\_g = 0; // [g] grams; quantity of available pollen on the specified day

UINT nectarVisitsToday = 0; // number of bees collected nectar from that patch today

UINT pollenVisitsToday = 0; // number of bees collected pollen from that patch today

//Updates daily\_data on the specified date

//function has to be overwrited

virtual void update(const date\_struct& date);

};

int patchType = 0;

int distanceToColony = 0;

int oldPatchID = 0;

int size\_sqm = 0;

int nectarConcFlowerPatch = 0;

int detectionProbability = 0;

int flightCostsNectar = 0;

int flightCostsPollen = 0;

int EEF = 0;

int danceCircuits = 0;

int danceFollowersNectar = 0;

int summedVisitors = 0;

int tripDuration = 0;

int tripDurationPollen = 0;

int mortalityRisk = 0;

int mortalityRiskPollen = 0;

int handlingTimeNectar = 0;

int handlingTimePollen = 0;

std::shared\_ptr<daily\_data> dailyData;

};

struct mite\_organiser : entity {

int workerCellListCondensed = 0;

int droneCellListCondensed = 0;

int cohortInvadedMitesSum = 0;

int invadedMitesHealthyRate = 0;

int invadedDroneCohortID = 0;

int invadedWorkerCohortID = 0;

};

meke\_enum(swarming\_type,

NO\_SWARMING,

SWARM\_CONTROL, // by beekeeper

PARENTAL\_COLONY,

DOUGHTER\_COLONY,

PRIME\_SWARM

);

struct model\_data {

//Model settings

UINT CRITICAL\_COLONY\_SIZE\_WINTER = 4000; // Martin (2001): 4000 adult workers during winter (from Free & Spencer-Booth 1958, Harbo 1983)

float MORTALITY\_EGGS = 0.03; // daily mortality rate of worker eggs (from Schmickl & Crailsheim (2007))

UINT HATCHING\_AGE = 3; // (days) age at which worker larvae hatch from eggs (from Winston (1987) p.50)

float MORTALITY\_DRONE\_EGGS = 0.064; // daily mortality rate of drone eggs (from Fukuda, Ohtani (1977))

UINT DRONE\_HATCHING\_AGE = 3; // (days) age at which drone larvae hatch from eggs (from Jay (1963), Hrassnig & Crailsheim (2005))

UINT DRONE\_PUPATION\_AGE = 10; // (days) age of pupation for drone (from Winston (1987))

UINT DRONE\_EMERGING\_AGE = 24; // age at which adult drones emerge from pupae (from Winston (1987))

UINT MAX\_EGG\_LAYING = 1600; // maximum egg laying rate per day

UINT EMERGING\_AGE = 21; // age at which adult workers emerge from pupae (Winston (1987) p.50)

float FORAGER\_NURSING\_CONTRIBUTION = 0.2; // contribution of foragers on brood care

float MAX\_BROOD\_NURSE\_RATIO = 3; // maximum amount of brood, nurse bees can care for (from Free & Racey (1968): 3; Becher et al. (2010): 2.65)

bool EGG\_LAYING\_IH = true; // if true: egg laying is affected by available nurse bees

UINT MAX\_BROODCELLS = 2000099; // maximum brood space

float DRONE\_EGGS\_PROPORTION = 0.04; // proportion of drone eggs (from Wilkinson & Smith (2002))

UINT SEASON\_START = 1; //edit // defines beginning of foraging period

UINT SEASON\_STOP = DAYS\_IN\_YEAR; //edit // end of foraging period & latest end of drone production

UINT PRE\_SWARMING\_PERIOD = 3; // defines period during which colony prepares for swarming (from Schmickl & Crailsheim (2007): 3 days, Winston (1987) p. 184: "until the week before swarming")

float ENERGY\_HONEY\_per\_g = 12.78; // [kJ/g] energy content of 1g honey (from USDA: 304kcal/100g (http://www.nal.usda.gov/fnic/foodcomp/search/))

//Process settings

bool ClearOnDead = false; // if true: some data will delete

bool QueenAgeing = false; // if true: egg laying rate decreases with queen age (following BEEPOP) and the queen is replaced every year

swarming\_type Swarming = swarming\_type::NO\_SWARMING

//Model data

std::list<egg\_group> egg\_groups;

std::list<egg\_drone\_group> egg\_drone\_groups;

std::list<larva\_group> larva\_groups;

std::list<larva\_drone\_group> larva\_drone\_groups;

std::list<pupa\_group> pupa\_groups;

std::list<pupa\_drone\_group> pupa\_drone\_groups;

std::list<drone\_group> drone\_groups;

std::list<in\_hive\_bee\_group> in\_hive\_bee\_groups;

std::list<forager\_squadron> forager\_squadrons;

std::list<flower\_patch> flower\_patchs;

std::list<mite\_organiser> mite\_organisers;

date\_struct date; // all other data in model\_data object setted for the end of this date (for the evening)

int Queenage = 1; // in days (min value is 1)

UINT SwarmingDay = 0; // day when swarming takes place; reset to 0 at the end of a year

double honeyEnergyStore = 0; // in [kJ] (kilojoules) (honey store of the colony (recorded in energy not in weight))

UINT PhoreticMites = 0; // all phoretic mites, healthy and infected

prop<bool> colonyDied;

std::string deathReason;

UINT TotalForagers = 0; // number of all foragers of the colony

UINT TotalWorkerAndDroneBrood = 0; // number of all worker and drone eggs, larvae and pupae

UINT TotalIHbees = 0; // number of in-hive bees present in the colony

model\_data(const model\_data&) = default;

model\_data(model\_data&&) = default;

model\_data& operator=(const model\_data&) = default;

model\_data& operator=(model\_data&&) = default;

static model\_data create\_sample();

//Gets Foraging Period For Today (using date field) in seconds

virtual int getForagingPeriodForToday();

};

struct internal\_model\_data;

struct model {

//sets sample model\_data

model();

model(std::shared\_ptr<model\_data> data);

const model\_data& get\_data();

daily\_step();

private:

std::unique\_ptr<internal\_model\_data> idata;

};

} // namespace BeeModel

} // namespace My

#endif // \_\_BEEMODEL\_H\_\_