Small Summery

Boris Prochnau

23. April 2014

1 Pseudocode

One thing to mention is that the Pseudocode snippets are marked as „Algorithm“, but in fact they are just functions and I dont know yet how to change the name in this particular LaTeX environment. Also I use a „=“ instead of „←“ because I think it provides better readability.
I separated this section in 3 parts, each one is dedicated to one of the functions used in EvolutionStep() below. It is possible to read
We will encounter 2 classes called „Trait“ and „Events“.

Attributes in a Trait Object:
class Trait

- BirthRate
- TotalBirthRate
- DeathRate
- TotalDeathRate
- TotalTraitRate
- TotalEventRate - [static]
- CompDeathRate[i][j] - [static]

Attributes in a Events Object:
class Events

- Dice
- EventTimes[i]
- ChosenTrait[i]
- isBirth[i]

Algorithm 1 EvolutionStep()

Require: -
Ensure: A full evolution Step happened

```
calculateEventRates();
sampleEventTime();
changeATrait();
```

This function does a full evolution step.

1.1 Calculating total-event rates

Algorithm 2 calculateEventRates()

Require: -
Ensure: All (total)Rates will be set

```
for i=0 to n-1 do
    calculateTotalDeathRateOf(i)
end for
calculateTotalBirthRates(0);
calculateTotalEventRate();
```

Algorithm 3 calculateTotalBirthRates(StartIndex: i)

Require: int i
Ensure: Total birthrate of Trait „i“ will be set (recursively)

1: Trait[i].TotalBirthRate = (Trait[i].Members)·(Trait[i].BirthRate)
2: **if** $i < n - 1$ **then**
3: calculateTotalBirthRates(i+1)
4: Trait[i].TotalBirthRate += $\frac{Trait.Mutation}{2} \cdot Trait[i+1].TotalBirthRate$
5: **end if**
6: **if** $i > 0$ **then**
7: Trait[i].TotalBirthRate += $\frac{Trait.Mutation}{2} \cdot Trait[i-1].TotalBirthRate$
8: **end if**

In Algorithm 3 in line 3 is used recursion, because this improves the calculation speed a lot, although it slightly makes code less intuitive.

Algorithm 4 calculateTotalDeathRateOf(TraitIndex: i)

[H]
Require: int i
Ensure: Total deathrate of Trait „i“ will be set

```
Trait[i].TotalDeathRate = 0;
addTotalIntrinsicDeathRateOf(i);
addTotCompetitionDeathRateOf(i);
```

Algorithm 5 addTotalIntrinsicDeathRateOf(TraitIndex: i)

1: Trait[i].TotalDeathRate = (Trait[i].DeathRate) · (Trait[i].Members)

Algorithm 6 addTotalCompetitionDeathRateOf(TraitIndex: i)

```
for j=0 to n-1 do
    Trait[i].TotalDeathRate += (Trait.CompDeathRate[i,j])·(Trait[j].Members);
end for
```

Algorithm 7 calculateTotalEventRate()

Require: -
Ensure: Current Totaleventrate is set

```
for i=0 to n-1 do
    Trait[i].TotalTraitRate = Trait[i].TotalBirthRate
                              + Trait[i].TotalDeathRate;
    Trait.TotalEventRate += Trait[i].TotalTraitRate;
end for
```

1.2 Sampling the next event-time

Here will appear a, not yet mentioned, object that will not be explained further, called Dice. The Dice Object will provide a uniform or exponential random Variable.

Algorithm 8 sampleEventTime()

Require: -
Ensure: First ringing Eventclock has been sampled

```
double Parameter = Trait.TotalEventRate;
double newEvent = this.Dice.RollExpDice(Parameter);
Events.EventTimes.push(newEvent);
```

Here we use Dice.RollExpDice(λ) to get $X \sim exp(\lambda)$. The same is possible for Dice.RollUnifDice(λ) to get $X \sim Unif[0, \lambda]$.

1.3 Changing a trait

Algorithm 9 changeATrait()

Require: -
Ensure: make a change to the Population with current Parameters

```
choseTraitToChange();
choseEventType();
executeEventTypeOnTrait();
```

Algorithm 10 choseTraitToChange()

Require: -
Ensure: Trait is chosen for changing

```
double Parameter = Trait.TotalEventRate;
double HittenTrait = Dice.rollUnif(Parameter);
for i = 1 to n-1 do
    if HittenTrait ≤ Trait[i].TotalEventRate then
        this.ChosenTrait.push(i);
        break;
    end if
    HittenTrait -= Trait[i].TotalEventRate
end for
```

Algorithm 11 choseEventType()

Require: -
Ensure: Decision for Birth or Death is made

```
int i = Events.ChosenTrait.lastentry();
double EventType = Dice.rollUnif(Trait[i].TotalTraitRate);
if EventType ≤ Trait[i].TotalBirthRate then
    Events.isBirth.push(true);
else
    Events.isBirth.push(false);
end if
```

Algorithm 12 executeEventTypeOnTrait()

Require: -
Ensure: Chosen event will occur on chosen trait

```
if isBirth then
    Trait[ChosenTrait.lastentry()] += 1;
else
    if ChosenTrait.Members > 0 then
        Trait[ChosenTrait.lastentry()] -= 1;
    end if
end if
```

List of Algorithms