The Impact of Resource Regeneration on Population Dynamics and Wealth Inequality: A Comparison of Sugarscape 1 and Sugarscape 3

Student number: 23212203

1. Introduction

Limited resources and endless desires. Throughout human history, resource consumption has been studied in multiple fields. The Sugarscape, an agent-based computational model designed by Epstein and Axtell (1996), to explore social phenomena from a bottom-up perspective. The core idea is to create a simplified environment populated by agents with simple rules and attributes and observe how complex social patterns emerge from their interactions. The model is used to study various aspects of social science, including resource distribution, trade, and wealth inequality. This observation in Sugarscape reflects real-world interactions between resource availability and social dynamics. Motesharrei et al. (2016) describe how the Earth System (resource regeneration) and the Human System (consumption) are connected, similar to how sugar regeneration and agent behavior work in the model.

This study aims to analyse how the regeneration rate of sugar resources affects population dynamics and wealth inequality. The investigation compares Sugarscape 1, the foundational model, with Sugarscape 3, an extended framework that integrates sugar regeneration rate and an aging mechanism to provide an understanding of how population changes and wealth is divided in simulation in different scenarios.

2. Methodology

The Sugarscape model simulates an artificial society on a 50*50 grid, where agents are randomly distributed across the environment. Agents, representing individuals, have attributes such as metabolism and vision, which enable them to navigate toward the highest sugar patch within their line of sight for survival. Agents can see and move in four directions (up, down, left, and right) within their vision range, with higher vision allowing them to access the patches with higher sugar. The metabolism mechanism operates in an order as agents consume sugar in each step. They will perish if their sugar is insufficient to move to the next desired patch.

In Sugarscape 1, agents follow basic movement rules to maximise sugar intake, and sugar fully regenerates at each time step. This provides unlimited resources for agents to consume. In Sugarscape 3, agents follow the same movement rules; however, the environment introduces a variable sugar regeneration rate for each patch. An aging mechanism is implemented, when an agent's age increases by one with each movement. Once an agent reaches its maximum age, it dies and is replaced by a new agent starting at age 1 from a random patch. This process allows the Sugarscape 3 model to maintain a stable population over time.

This study compares population dynamics and wealth inequality between Sugarscape 1 and Sugarscape 3. To achieve this, this study adjusts the sugar regeneration rate in 1, 2, 3, and maximum and sets the initial populations of 500 and 1000 in both models and adds the Gini index(0 is equality, 1 is inequality) in Sugarscape 1 to measure the degree of wealth inequality in multiple scenarios. The agent's settings are kept the same with the initial sugar, metabolism, and vision randomly assigned within specified ranges to analyse how different resource regeneration rates and population sizes impact wealth distribution. A detailed parameter and mechanism setting is provided in **Table 1**.

In BehaviorSpace, this study analyses the agent survival rate, mean metabolism, mean vision, and the Gini index in Sugarscape 1. However, since the population remains constant in Sugarscape 3, calculating the agent survival rate is not meaningful. Instead, it can calculate the mean agent age to gain further insights. Additionally, the number of repetitions is set to 30 to minimise randomness, and the time limit is set to 3,000 ticks to avoid a warm up period. All details are provided in **Table 2**.

Table 1: Sugarscape 1 and Sugarscape 3 new setting

Parameter/ mechanism	Sugarscape 1	Sugarscape 3
Initial population	500, 1,000	500, 1,000
Sugar regeneration rate	1, 2, 3, max	1, 2, 3, max
Initial agent's sugar	5-25	5-25
Agent's metabolism	1-4	1-4
Agent's vision	1-6	1-6
Agent's age	-	0 to 60-100
Agent's regeneration	-	Yes

Table 2: Sugarscape 1 and Sugarscape 3 BehaviorSpace measurement

Measurement/ setting	Sugarscape 1	Sugarscape 3
Agent survival rate	Number of agents/ Initial population	-
Agent's metabolism	Mean metabolism of agents	Mean metabolism of agents
Agent's vision	Mean vision of agents	Mean vision of agents
Agent's age	-	Mean age of agents
Gini index	0-1	0-1
Repetitions	30	30
Time limit	3000	3000

3. Results

The following analysis in Sugarscape 1 (**Figure 1)** comparing the initial population settings between 500 and 1000, with the parameters of survival rate, metabolism, vision, and Gini index.

1. In the survival rate, a higher sugar regeneration rate leads to increased survival, which is expected as greater resource availability enhances agents' chances of survival. The initial population of 500 exhibits a higher survival rate across all sugar regeneration rates of 1000, intuitively suggesting that fewer agents competing for the same resources results in higher survival.

2. In the metabolism, A positive correlation with sugar regeneration rate is observed,

meaning agents tend to have higher metabolism in resource-abundant environments. Also, metabolism is slightly lower in the initial population of 1000 than in 500, showing that with a larger population, lower metabolism may be necessary for agents to sustain under the same resource conditions.

3. There is no clear pattern across different sugar regeneration rates and the vision.

4. Higher sugar regeneration rates correspond to an increase in the Gini index, indicating that resource abundance amplifies wealth inequality.

In summary, greater resource availability enhances agent survival but increases wealth inequality. When resources remain constant, a smaller population experiences higher survival rates and metabolism than a larger population.

Figure 1: Comparison of Initial Population Settings in Sugarscape 1

The scatter plots (**Figure 2**) reveal positive correlations between the Gini index and metabolism and survival rate, but the correlation with vision is weaker in Sugarscape 1. In detail, for a population of 500, Gini index aligns with higher metabolism (r = 0.91) and survival rate (r = 0.81), and vision shows a weak negative correlation (r = -0.12). In a population of 1,000, these correlations strengthen slightly (metabolism: r = 0.933, survival: r = 0.91), and vision remains a weak correlation (r = 0.21).

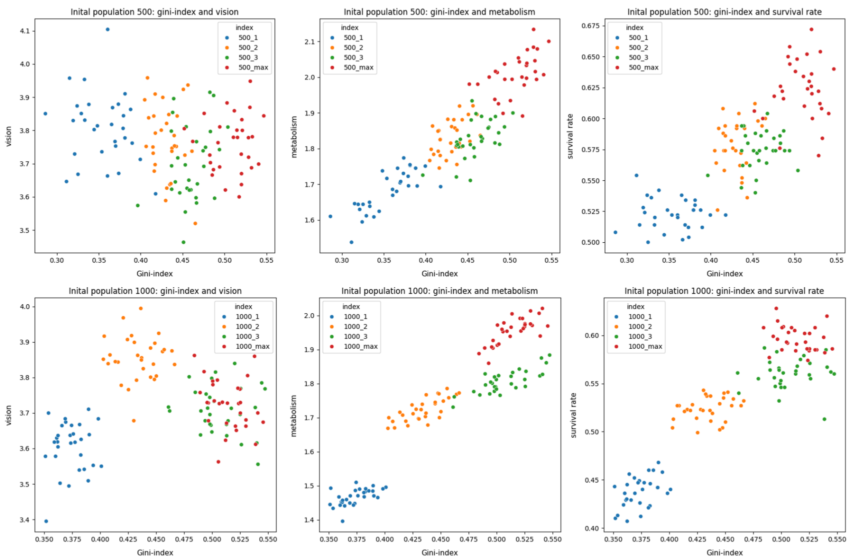


Figure 2: Correlation Coefficient of Gini-index in Sugarscape 1

In Sugarscape 3 (**Figure 3)**, regardless of population size 500 or 1000, the Gini index strongly correlates with metabolism (r = 0.87, 0.89). Its correlation with age is lower in a population of 500 (r = 0.57) but increases to 0.87 in a population of 1000. Vision maintains a weak correlation (r < 0.4) in both cases.

Overall, Sugarscape 1 and 3 show strong positive correlations between the Gini index and metabolism. While vision's correlation with wealth inequality was weak or negative in Sugarscape 1, it becomes consistently positive in Sugarscape 3, though it remains weak.

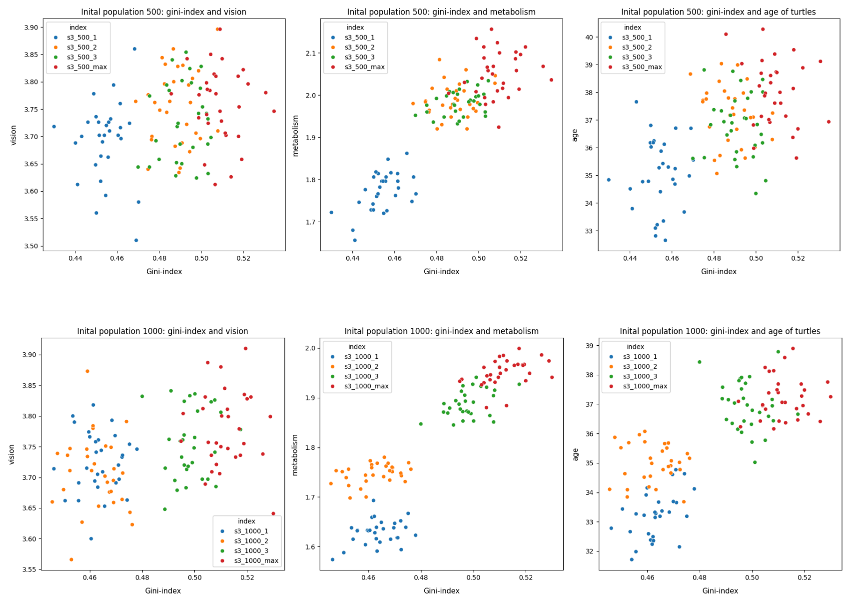


Figure 3: Correlation Coefficient of Gini-index in Sugarscape 3

We can discover in **Figure 4** that both Sugarscape 1 and 3 exhibit higher wealth inequality with increasing sugar regeneration rates. As resource availability increases, the difference between the two models diminishes.

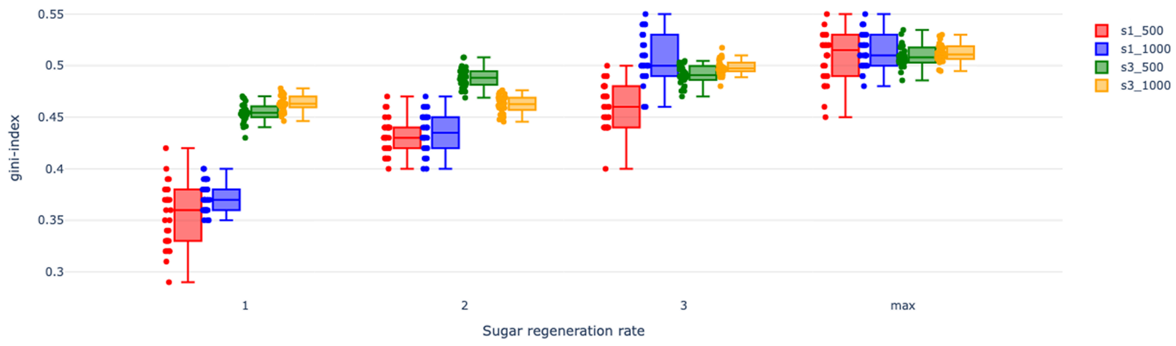


Figure 4: Compare the Gini index between Sugarscape 1 and 3

Lastly, we ran Sugarscape 3 for 30 iterations over 3,000 ticks, plotting the median trend line **(Figure 5)**. The shaded area represents the interquartile range (first to third quartile). At regeneration rates of 1, 3, and Maximum, the Gini index for a population of 1,000 is slightly higher than for 500. Interestingly, at a regeneration rate of 2, the Gini index for a population of 500 is significantly higher than for 1,000.

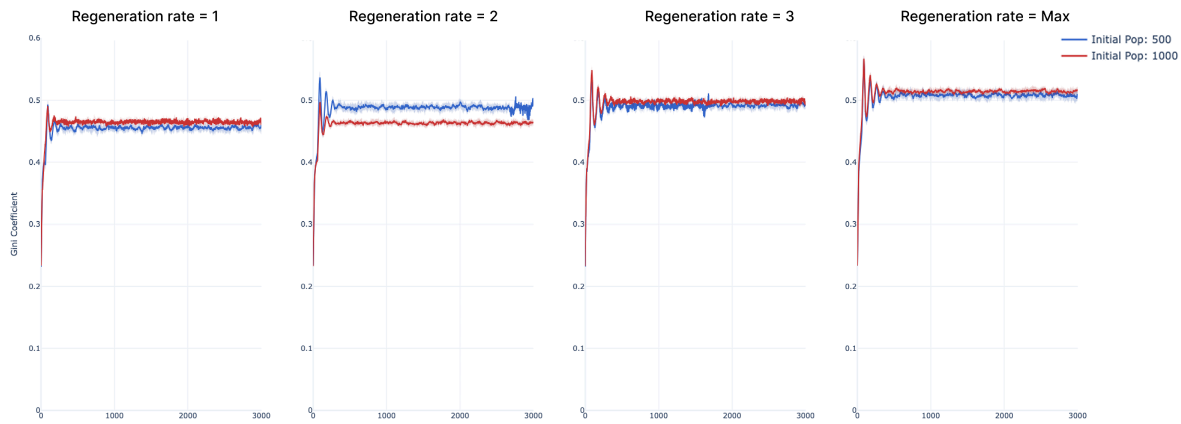


Figure 5: Gini index Over Time at Different Regeneration Rates in Sugarscape 3

4. Discussion and Conclusion

This study examined how sugar resource regeneration affects population dynamics and wealth inequality by comparing Sugarscape 1 and Sugarscape 3 in different settings. Higher regeneration rates increase survival and amplify inequality in Sugarscape 1. In both models, metabolism strongly correlates with the Gini index, indicating that greater metabolic demands drive wealth disparity. While vision had a weak or negative correlation with inequality in Sugarscape 1, it became consistently positive in Sugarscape 3, though still weak, suggesting a secondary role in wealth distribution.

Both models show that resource abundance intensifies inequality and Sugarscape 3 provides a more dynamic representation of real-world socioeconomic systems, where generational turnover and finite lifespans influence wealth distribution. However, real societies are more complex. Advanced Sugarscape models introduce dual-commodity economies, allowing trade between sugar and spice (Epstein & Axtell, 1996). Motesharrei et al. (2016) show that resource or labor shortages can trigger societal collapse. Rahman et al. (2008) found that population replacement enhances social mobility and eases wealth concentration, and inheritance can make agents who might be removed survive. Future research should explore additional mechanisms, such as inheritance and trade, to improve the model's real-world applicability.

5. References

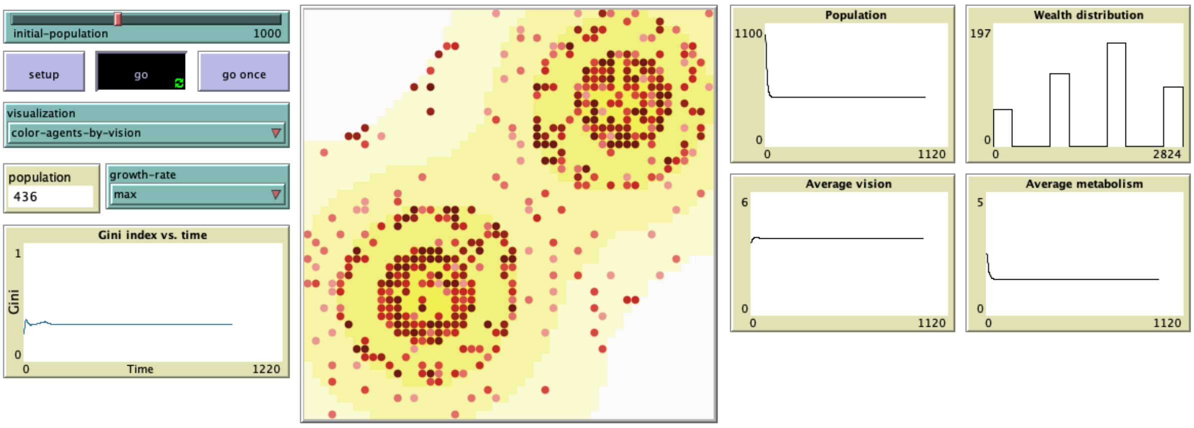
Epstein, J.M. and Axtell, R. (1996). Growing Artificial Societies: Social Science from the Bottom Up. MIT Press. Available at: https://direct.mit.edu/books/monograph/2503/Growing-Artificial-SocietiesSocial-Science-from

Motesharrei, S., Rivas, J., Kalnay, E., et al., (2016). Modeling sustainability: Population, inequality, consumption, and bidirectional coupling of the Earth and Human Systems. National Science Review, 3(4), pp.470–494. Available at: https://doi.org/10.1093/nsr/nww081

Rahman, Arash & Setayeshi, Saeed & Shamsaei, Mojtaba. (2008). An analysis to wealth distribution based on sugarscape model in an artificial society. IJE Transactions B: Applications. 20. Available at: https://www.ije.ir/article_71664_0d514c0116334fac86bff7d3e3d3d694.pdf

6. Appendix

Sugarscape 1:



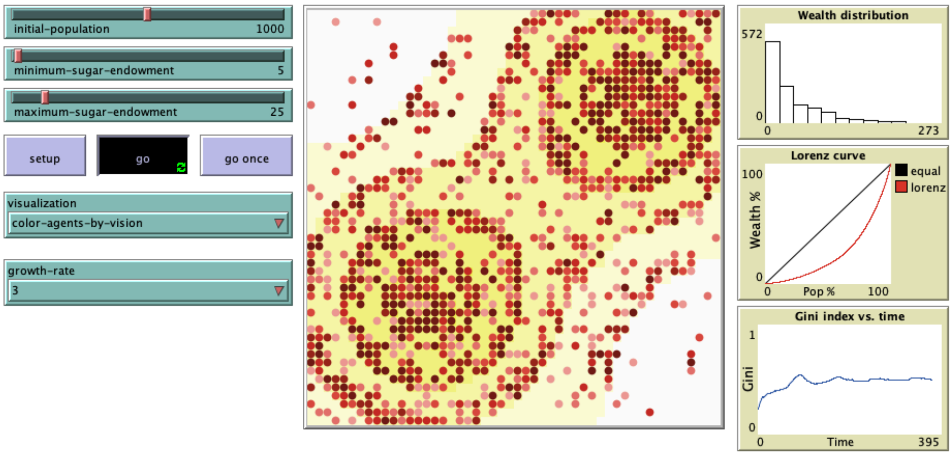

```
globals [
 gini-index-reserve  ;;
 lorenz-points       ;;
]

turtles-own [
 sugar         ;; the amount of sugar this turtle has
 metabolism      ;; the amount of sugar that each turtles loses each tick
 vision         ;; the distance that this turtle can see in the horizontal and vertical directions
```

```
 vision-points   ;; the points that this turtle can see in relative to it's current position (based on vision)
]

patches-own [
 psugar          ;; the amount of sugar on this patch
 max-psugar      ;; the maximum amount of sugar that can be on this patch
]

;;
;; Setup Procedures
;;

to setup
 clear-all
 create-turtles initial-population [ turtle-setup ]
 setup-patches
 update-lorenz-and-gini
 reset-ticks
end

to turtle-setup ;; turtle procedure
 set color red
 set shape "circle"
 move-to one-of patches with [not any? other turtles-here]
 set sugar random-in-range 5 25
 set metabolism random-in-range 1 4
 set vision random-in-range 1 6
 ;; turtles can look horizontally and vertically up to vision patches
 ;; but cannot look diagonally at all, only up, down, left, right
 set vision-points []
 foreach (range 1 (vision + 1)) [ n ->
   set vision-points sentence vision-points (list (list 0 n) (list n 0) (list 0 (- n)) (list (- n) 0))
 ]
 run visualization
end

to setup-patches
 file-open "sugar-map.txt"
 foreach sort patches [ p ->
   ask p [
     set max-psugar file-read
     set psugar max-psugar
     patch-recolor
   ]
 ]
 file-close
end

;;
;; Runtime Procedures
;;

to go
 if not any? turtles [
   stop
```

```
 ]
 ask patches [
   patch-growback
   patch-recolor
 ]
 ask turtles [
   turtle-move
   turtle-eat
   if sugar <= 0
     [ die ]
   run visualization
 ]
 update-lorenz-and-gini
 tick
end

to turtle-move ;; turtle procedure
 ;; consider moving to unoccupied patches in our vision, as well as staying at the current patch
  let move-candidates (patch-set patch-here (patches at-points vision-points) with [not any?
    turtles-here])
 let possible-winners move-candidates with-max [psugar]
 if any? possible-winners [
   ;; if there are any such patches move to one of the patches that is closest
   move-to min-one-of possible-winners [distance myself]
 ]
end

to turtle-eat ;; turtle procedure
 ;; metabolize some sugar, and eat all the sugar on the current patch
 set sugar (sugar - metabolism + psugar)
 set psugar 0
end

to patch-recolor ;; patch procedure
 ;; color patches based on the amount of sugar they have
 set pcolor (yellow + 4.9 - psugar)
end

to patch-growback ;; patch procedure

 if growth-rate != "max" [
   let growth-amount growth-rate
   set psugar min (list max-psugar (psugar + growth-amount))
 ]

 if growth-rate = "max" [
   set psugar max-psugar
 ]

 patch-recolor
end

;;
;; Utilities
;;
```

```
to-report random-in-range [low high]
 report low + random (high - low + 1)
end

to update-lorenz-and-gini
 let num-people count turtles
 let sorted-wealths sort [sugar] of turtles
 let total-wealth sum sorted-wealths
 let wealth-sum-so-far 0
 let index 0
 set gini-index-reserve 0
 set lorenz-points []
 repeat num-people [
   set wealth-sum-so-far (wealth-sum-so-far + item index sorted-wealths)
   set lorenz-points lput ((wealth-sum-so-far / total-wealth) * 100) lorenz-points
   set index (index + 1)
   set gini-index-reserve
     gini-index-reserve +
     (index / num-people) -
     (wealth-sum-so-far / total-wealth)
 ]
end


;;
;; Visualization Procedures
;;

to no-visualization ;; turtle procedure
 set color red
end

to color-agents-by-vision ;; turtle procedure
 set color red - (vision - 3.5)
end

to color-agents-by-metabolism ;; turtle procedure
 set color red + (metabolism - 2.5)
end
```

Sugarscape 3:




```
globals [
 gini-index-reserve
 lorenz-points
]

turtles-own [
 sugar          ;; the amount of sugar this turtle has
 metabolism     ;; the amount of sugar that each turtles loses each tick
 vision         ;; the distance that this turtle can see in the horizontal and vertical directions
 vision-points  ;; the points that this turtle can see in relative to it's current position (based on vision)
 age            ;; the current age of this turtle (in ticks)
 max-age        ;; the age at which this turtle will die of natural causes
]

patches-own [
 psugar         ;; the amount of sugar on this patch
 max-psugar     ;; the maximum amount of sugar that can be on this patch
 sugar-counter  ;; counter for partial sugar growth
]

;;
;; Setup Procedures
;;

to setup
 if maximum-sugar-endowment <= minimum-sugar-endowment [
      user-message "Oops: the maximum-sugar-endowment must be larger than the
    minimum-sugar-endowment"
   stop
 ]
 clear-all
 create-turtles initial-population [ turtle-setup ]
 setup-patches
 update-lorenz-and-gini
 reset-ticks
end

to turtle-setup ;; turtle procedure
 set color red
 set shape "circle"
```

```
 move-to one-of patches with [not any? other turtles-here]
 set sugar random-in-range minimum-sugar-endowment maximum-sugar-endowment
 set metabolism random-in-range 1 4
 set max-age random-in-range 60 100
 set age 0
 set vision random-in-range 1 6
 ;; turtles can look horizontally and vertically up to vision patches
 ;; but cannot look diagonally at all
 set vision-points []
 foreach (range 1 (vision + 1)) [ n ->
   set vision-points sentence vision-points (list (list 0 n) (list n 0) (list 0 (- n)) (list (- n) 0))
 ]
 run visualization
end

to setup-patches
 file-open "sugar-map.txt"
 foreach sort patches [ p ->
   ask p [
     set max-psugar file-read
     set psugar max-psugar
     patch-recolor
   ]
 ]
 file-close
end

;;
;; Runtime Procedures
;;

to go
 if not any? turtles [
   stop
 ]
 ask patches [
   patch-growback
   patch-recolor
 ]
 ask turtles [
   turtle-move
   turtle-eat
   set age (age + 1)
   if sugar <= 0 or age > max-age [
     hatch 1 [ turtle-setup ]
     die
   ]
   run visualization
 ]
 update-lorenz-and-gini
 tick
end

to turtle-move ;; turtle procedure
 ;; consider moving to unoccupied patches in our vision, as well as staying at the current patch
```

```
  let move-candidates (patch-set patch-here (patches at-points vision-points) with [not any?
    turtles-here])
 let possible-winners move-candidates with-max [psugar]
 if any? possible-winners [
  ;; if there are any such patches move to one of the patches that is closest
  move-to min-one-of possible-winners [distance myself]
 ]
end

to turtle-eat ;; turtle procedure
 ;; metabolize some sugar, and eat all the sugar on the current patch
 set sugar (sugar - metabolism + psugar)
 set psugar 0
end

to patch-recolor ;; patch procedure
 ;; color patches based on the amount of sugar they have
 set pcolor (yellow + 4.9 - psugar)
end

to patch-growback ;; patch procedure

 if growth-rate != "max" [
  let growth-amount growth-rate
  set psugar min (list max-psugar (psugar + growth-amount))
 ]

 if growth-rate = "max" [
  set psugar max-psugar
 ]

 patch-recolor
end

to update-lorenz-and-gini
 let num-people count turtles
 let sorted-wealths sort [sugar] of turtles
 let total-wealth sum sorted-wealths
 let wealth-sum-so-far 0
 let index 0
 set gini-index-reserve 0
 set lorenz-points []
 repeat num-people [
  set wealth-sum-so-far (wealth-sum-so-far + item index sorted-wealths)
  set lorenz-points lput ((wealth-sum-so-far / total-wealth) * 100) lorenz-points
  set index (index + 1)
  set gini-index-reserve
    gini-index-reserve +
    (index / num-people) -
    (wealth-sum-so-far / total-wealth)
 ]
end

;;
;; Utilities
```

```
;;

to-report random-in-range [low high]
 report low + random (high - low + 1)
end

;;
;; Visualization Procedures
;;

to no-visualization ;; turtle procedure
 set color red
end

to color-agents-by-vision ;; turtle procedure
 set color red - (vision - 3.5)
end

to color-agents-by-metabolism ;; turtle procedure
 set color red + (metabolism - 2.5)
end
```