Efectos poblacionales de la modificación genética

NetLogo

Karen Arteaga Mendoza
Javier Corral Lizárraga

Matemática Computacional 002
Mayo 2021
Profesor Juan Carlos Martínez García

Índice

Introducción

Resumen

La ingeniería genética es la técnica que se emplea para manipular genéticamente organismos con el fin de modificar sus características, cambiando el material genético.[1] Las técnicas de la ingeniería genética se han aplicado a diversos seres vivos, desde perros hasta tortugas, y han traído consigo varias contradicciones y debates por parte de la comunidad de la biotecnología. Sin embargo, la manipulación genética en humanos es un tema que, hasta el día de hoy, causa controversia para la mayor parte de la comunidad científica. En 2015, cuando se celebró la primera reunión de la Cumbre Internacional sobre la Edición Genética en Humanos, se llegó a la conclusión de que sería irresponsable hacer modificaciones al código genético de los humanos en células germinales; es decir, no se debe editar genéticamente al humano en las células que pueden transmitir su información genética. [2] Es por eso que en este estudio se busca desarrollar una simulación de los efectos de la modificación genética para la descendencia de una población acotada con un genoma simplificado.

El objetivo de este estudio es desarrollar un sistema que simule la interacción de una población de humanos no modificados genéticamente al introducir una población de humanos modificados genéticamente, los cuales pueden heredar la edición de sus genes. Se busca crear una aplicación, amigable con el usuario, en lenguaje NetLogo que experimente los diferentes posibles escenarios sobre la simulación para obtener perspectivas de los potenciales efectos que causaría la introducción de individuos modificados genéticamente sobre una población humana caracterizada.

Después de haber delimitado el genoma humano de la muestra poblacional a 3 genes (fertilidad, fuerza y defensa), se especifican las interacciones que existen entre los individuos de la población (reproducción y pelea). Finalmente, se genera el algoritmo que permite la introducción de los individuos con material genético modificado.

Tras una cantidad representativa de pruebas en la simulación, se llegó a la conclusión de que los genes de los individuos editados se esparcen por la población en un número reducido de generaciones. La modificación genética se hereda a través de la descendencia de la poblacion en materia de tan solo algunas generaciones e influye en el promedio global de cada rasgo que tenía modificado de forma permanente y significativa.

Es claro que se necesitan simulaciones mucho más precisas del ser humano y de su interacción poblacional para poder hacer declaraciones sobre la bondad de la modificación genética, pero podemos incluso desde una simulación tan primitiva concluir que el efecto de la modificación genética sobre una población tan interconectada puede tener efectos que lleguen hasta el último individuo de la sociedad y que se debe tener cuidado al trabajar con estos para limitar el potencial riesgo.

Palabras clave

Ingeniería genética, NetLogo, modificación genética, simulación, población

Desarrollo

Antecedentes

El Juego de la Vida

Es un juego de cero jugadores, cuya evolución está determinada por el estado inicial y no necesita ninguna entrada posterior realizada por el jugador.

El estado del tablero evoluciona respecto al estado de las células y sus vecinos que pueden estar vivos o muertos.

Tiene las siguientes reglas:

1. Una célula muerta con exactamente 3 células vecinas vivas "nace" (es decir, al turno siguiente estará viva).
2. Una célula viva con 2 o 3 células vecinas vivas sigue viva, en otro caso muere (por "soledad" o "superpoblación").

El juego fue diseñado por el matemático John Conway en 1970 para una publicación en la revista *Scientific American*, tras su lanzamiento atrajo mucho interés debido a que era un buen ejemplo de los fenómenos de emergencia y autoorganización. Tras el trabajo extensivo de la comunidad se descubrieron patrones de celulas que permiten simular comportamientos complejos, tanto que se demostró que el mismo juego es turing completo. [6]

Tomamos este juego como base debido a su modelo de interacción entre individuos, estos solo se pueden dar de forma discreta entre 2 entidades a la vez que están adyacentes entre sí dentro de un tablero de medidas también discretas, simplificando nuestro modelo a solo el cálculo de la interacción de cada individuo con cada uno de sus potenciales 8 vecinos.

A pesar de que el juego de la vida ya presenta comportamientos complejos a pesar de sus reglas simples (solo tienen el estado de vivo o muerto), decidimos crear distintos tipos de individuos, los cuales tienen a su vez rasgos determinados por una serie de sistemas de interacción.

NetLogo

NetLogo es un ambiente de programación utilizado para modelar programas basados en agentes diseñado por Uri Wilensky en 1999, está basado en los lenguajes de programación Java y Scala y es un software libre de código abierto y público. NetLogo se usa principalmente para desarrollar sistemas que se alteran con el tiempo como fenómenos sociales. [5] Netlogo se ejecuta como una aplicación de escritorio, aunque también existe la

versión Web, se abren una línea de comandos llamada *Terminal de instrucciones* y una cuadrícula negra que reflejará los comandos dictados. NetLogo también admite el código escrito en la pestaña *Código* y botones, barras, cuadros de texto, etc. para hacer el programa más eficiente y amigable con el usuario. (Ver imagen 1)

El lenguaje de NetLogo se compone de agentes que siguen instrucciones. Hay 4 tipos generales de agentes: *turtles, patches, links y observer*. Los *turtles* se pueden mover a lo largo de la cuadrícula y los *patches* son los cuadros de la cuadrícula. Las instrucciones se dan en bloques llamados procedimientos, que se escriben en código empezando con la palabra *to* seguida del nombre del procedimiento (i.e. *to grow*). NetLogo también permite delimitar especies (*breed*) que responden a rasgos especificados en el código. [6]

NetLogo, en su versión 6.2.0, es la herramienta que se utilizó para modelar el proyecto, esto facilitó el programa, pues lo hizo más accesible y eficiente a la hora de trabajar con los gráficos y el modelado basado en agentes de una forma estandarizada diseñada justo para este tipo de proyectos.

Implementación

Caracterización de la población

En la población, existen dos tipos de individuos, los bebés y los adultos. Cada adulto puede ser hombre o mujer y está caracterizado por 5 rasgos: edad, fertilidad, fuerza, defensa y modificación genética. Cada año aumenta la edad de los adultos. La edad delimita las actividades que puede hacer un individuo en la población: pelear con dos o más años, reproducirse con menos de 4 años en el caso de las mujeres y hacerse ancianos con más de 8 años. La fertilidad, fuerza y defensa se miden con valores de 1-10 y describen qué tan fértil, fuerte y capaz de defenderse es cada individuo. La modificación genética describe si el individuo fue o no modificado genéticamente por herencia de sus padres o por intervención externa. Los bebés no tienen sexo ni edad y tienen 4 rasgos: fertilidad heredada, fuerza heredada, defensa heredada y si son o no modificados genéticamente.

Cada individuo tiene vecinos, que son los individuos que viven en los 8 cuadros alrededor, con los que interactúa. Las interacciones principales son pelear y reproducirse. Las peleas se dan entre adultos mayores de 2 años y para ellas se toman en cuenta los rasgos fuerza y defensa. Si la defensa y la fuerza del vecino son menores, entonces el vecino muere con 80% de probabilidad. Si la fuerza del vecino es menor, pero la defensa es mayor, entonces el vecino muere con 50% de probabilidad.

La otra interacción que hay con los vecinos es la reproducción. Para la reproducción se necesita un hombre y una mujer de edad menor a 4 años (*ticks*) que juntos sumen una fertilidad mayor a 10. La herencia genética de los padres difiere si alguno de ellos fue modificado genéticamente. En caso de que uno de los dos sí haya sido modificado, el bebé recibirá los genes con mayor valor del padre o la madre. Es decir, si el padre tiene 10 de fertilidad, 8 de defensa y 3 de fuerza, pero la madre tiene 10 de fertilidad, 7 de defensa y 10 de fuerza, entonces el bebé heredará 10 en fertilidad del padre o madre, 8 en defensa del padre y 10 en fuerza de la madre; en este caso, el hijo habrá sido modificado genéticamente. Para el caso en que ninguno de los dos haya sido modificado, la herencia genética será, en cada uno de los rasgos, la suma de la mitad del padre y la madre. Es decir, en el ejemplo anterior, el bebé no modificado heredaría 10 en fertilidad (5 del padre más 5 de la madre), 7.5 en defensa (4 del padre y 3.5 de la madre) y 6.5 de fuerza (1.5 del padre y 5 de la madre).

Los bebés no interactúan con los adultos, pero guardan en sus atributos la información genética de sus padres. Cuando pasa un año, el bebé se vuelve adulto y asigna esta información genética a los atributos de adulto. El bebé adquiere el sexo aleatoriamente cuando es adulto y su edad es 1.

La población máxima es de 5,041 individuos delimitados por una cuadrícula de 71x71 recuadros, pero, para que se puedan dar las siguientes generaciones con suficiente espacio, se toman medidas que estabilizan la cantidad de individuos vivos por año (*tick*). En primer lugar, cada año muere, aleatoriamente, alrededor de 3% de la población total. En segundo lugar, si la densidad de población excede el 60%, entonces habrá un recorte masivo y se morirá alrededor del 50% de la población viva.

Inserción de individuos modificados genéticamente

La inserción de individuos genéticamente modificados se maneja a través de la interacción con el usuario. A este se le da la capacidad de determinar los valores que tendrá cada atributo de su nueva generación y puede decidir cuándo quiere insertarla en la simulación y cuántos de estos individuos se quieren insertar. Estos nuevos individuos se insertan sobre los *patches,* que son los cuadros de la cuadrícula, no utilizados por otros individuos de manera uniforme dentro de la parrilla donde viven. Los individuos modificados son detectables por el color: las mujeres son de color amarillo y los hombres de color azul claro (cyan). Cuando haya reproducción entre individuos modificados genéticamente, los bebés serán de color naranja.

Partes notables del programa

El código se realizó en el lenguaje NetLogo y se puede encontrar completo en la sección de anexos. La población del programa se divide en tres agentes que extienden del agente *turtle* de NetLogo. Estos agentes son hombres (*men),* mujeres (*women)* y bebés (*babies)*. Los agentes hombres y mujeres tienen 5 atributos: edad (*age*) de tipo *number*, fertilidad (*fertility*) de tipo *number*, fuerza (*strength*) de tipo *number*, defensa (*defense*) de tipo *number* y modificación genética (*modif*) de tipo *boolean*; y los bebés tienen 4: fertilidad heredada (*inhF*) de tipo *number*, fuerza heredada (*inhS*) de tipo *number,* defensa heredada (*inhD*) *number* y modificación genética (*modif*) de tipo *boolean*.

La cantidad de hombres y de mujeres surge en la primera generación de acuerdo con el marcador de densidad de población que da el usuario. La delimitación de la población está marcada por una cuadrícula de 71 x 71 unidades; es decir, el máximo de población es 5,041 habitantes. Los habitantes surgen en la cuadrícula de manera aleatoria y se dividen entre hombres y mujeres una relación 50% hombres y 50% mujeres con índices de fertilidad, fuerza y defensa aleatorios para cada individuo. Además, la población también reconoce a los vecinos, que son los hombres, mujeres o bebés que habitan en los 8 cuadros alrededor de un agente.

El programa está separado en 5 procedimientos clave: *birth, grow, fight, reproduce y killPob.* Cada uno de estos procedimientos permiten que la población se desarrolle de acuerdo con las leyes expuestas en la caracterización de la población. En cambio, el procedimiento *insertarMod* genera el cambio que representa a la mutación genética que permite el desarrollo de los individuos mutados. Los procedimientos se acceden desde los botones que se presentan en la interfaz de la aplicación (ver imagen _ en Anexos).

Al seleccionar el botón *go* se ejecutan los procedimientos en orden: *grow, fight, reproduce, killPob*. El primer procedimiento (*grow*) añade un año a la edad de los agentes hombres y mujeres y cambia el color a gris de aquellos agentes que sean mayores a 8 años. Después, convierte a todos los bebés que existan en la población en agentes hombres o mujeres aleatoriamente, asigna su fertilidad, fuerza y defensa como la mitad de la sumas heredadas y la edad igual a 1. Finalmente, cambia el color de los nuevos hombres a azul y el de las nuevas mujeres a rojo. El segundo procedimiento es *fight*, que indica a todos los habitantes hombres mayores de 2 años a pelear contra sus vecinos y luego a todos los agentes mujeres mayores de 2 años a pelear contra sus vecinos. Las peleas tienen como resultado que

los vecinos mueran o vivan de acuerdo con las reglas expresadas en la sección *Caracterización de la población.*

El procedimiento *reproduce* describe la forma en la que surgen los bebés mutados y no mutados en el programa. Para la creación de un bebé se necesita un padre y una madre, con edad menor a 4, vecinos que juntos tengan una suma de fertilidad mayor a 10. La herencia genética se da dependiendo de la modificación genética de los padres. Si alguno de los dos fue modificado genéticamente, entonces el bebé heredará los máximos de fertilidad, fuerza y defensa entre padre y madre. En cambio, si ninguno de los dos es modificado genéticamente, el padre y la madre heredan la mitad de su fertilidad, fuerza y defensa a sus bebés. Los bebés surgen en la cuadrícula con el procedimiento *birth*, éste asigna los atributos guardados en la lista a cada bebé y los esparce por la cuadrícula a los bebés de forma aleatoria. Los bebés tienen forma de círculo sin relleno (*circle 2*) y son de color verde limón.

El procedimiento *killPob* implementa dos procesos, el primero es hacer desaparecer o matar (*kill*) a todos los habitantes hombres y mujeres que tengan color gris, y el segundo es llevar el control de la sobrepoblación. Para tratar la sobrepoblación se implementaron 2 medidas, la primera elimina alrededor del 9% de la población total de individuos en cada tick y la segunda elimina al 50% de la población cuando hay más de 60% de densidad de población. Finalmente, el procedimiento *insertarMod* recibe un argumento de tipo entero que delimita el número de individuos modificados que se agregarán a la población total (*cuantos*). La modificación genética se da en las variables *Nueva-Fuerza*, *Nueva-Fertilidad* y *Nueva-Defensa* que describen los valores que se insertarán a los individuos modificados en cada atributo. Después, se generan aleatoriamente la cantidad especificada de hombres o mujeres genéticamente modificados que son de color azul claro y amarillo respectivamente.

Resultados

Aplicación

La aplicación está diseñada para ser amigable con alguien que no esté familiarizado ni con la plataforma, ni con el tema de la modificación genética. Esta se muestra con valores predeterminados que arrojaran una simulación inicial interesante, pero también le ofrece al usuario la capacidad de modificar ciertos parámetros para que pueda experimentar por sí mismo. La aplicación a su vez se actualiza con cada *tick* que pasa, que en la simulación representa un año, esto le permite al usuario visualizar con las gráficas que se actualizan y la

población que está bien diferenciada dentro de la parrilla donde viven la forma en que evoluciona su comportamiento y sus atributos.

La interfaz de la aplicación tiene una cuadrícula negra, a su derecha un *slider* y 4 botones azules que sirven para la ejecución de los *ticks*, abajo dos gráficas y, bajo la cuadrícula, otra serie de botones y *sliders* que sirven para insertar a la población genéticamente modificada. El usuario puede utilizar cada uno de estos botones para diseñarla población que guste y estudiar su evolución en las gráficas de los lados.

Dificultades en la implementación

NetLogo es un lenguaje que, por ser de *software* libre, tiene una comunidad pequeña a diferencia de otros lenguajes, lo cual significó que no había tanto soporte por la comunidad. A pesar de esto, tras leer la documentación que estaba bien explicada, pudimos resolver todos los problemas que se presentaron. Algunos casos notables de limitaciones del lenguaje con respecto a otros serían el que el *scope* de las variables era difícil de determinar, lo cual nos orilló a declarar listas globales con variables que pudiéramos utilizar dentro del *scope* de un procedimiento, por otro lado. También hubo dificultades en la adaptación al formato en que se escribe lisp (el lenguaje en que está basado NetLogo), lo cual, antes de un consenso en cómo se debía escribir, tomó mucho tiempo y esfuerzo intentar entender lo que hacía el código.

En cuanto a la traducción del problema a un lenguaje de programación, hubo dificultades para tratar a la población. El control de la sobrpoblación causó varios problemas porque no se conocía un algoritmo que estabilizara la población que nace y la que muere. Esto llevó a que hubiera un desequilibrio y pasaba uno de dos casos, la cantidad de individuos superaba la cantidad de patches en la cuadrícula, o la cantidad de individuos tendía a cero y no sobrevía la población. Para resolverlo se encontraron dos soluciones que controlan a la población. En primer lugar, se hizo un procedimiento que matara alrededor del 9% de la población cada año y, en segundo lugar, se implementó una medida de control de sobrepoblación que mata al 50% de la población viva en el caso en que la densidad sobrepase 60%. Estas dos medidas estabilizaron la población de tal manera que se pudo llegar a una cantidad ilimitada de generaciones si que hubiera una tendencia a infinito o a cero.

Limitaciones conocidas

La simulación permite que el usuario genere una población con los datos que el usuario da. Sin embargo, hay complicaciones cuando el usuario lo hace de manera irresponsable, ya que

cabe la posibilidad de que el usuario inocentemente intente trabajar con valores muy grandes que demanden de los recursos del lenguaje y de su computadora más de los que puede proveer, terminando en un *softlock* del cual solo se puede reanudar la interacción yendo al menú de NetLogo y pidiéndole que pare la ejecución del programa con el botón *halt*.
Se intentó acotar esta posibilidad y demás *bugs* solo proveyendo al usuario con *sliders* que tienen intervalos de valores válidos para el programa, pero debido a la multiplicidad de dispositivos y *hardware* en que se puede ejecutar, no se limitaron todos los valores que podrían dejar en *softlock* a usuarios con hardware no adecuado.

Manual de usuario

Al ejecutar el programa, el usuario se encontrará con una cuadrícula negra, seguida de diferentes botones azules, *sliders* verdes y recuadros para gráficas (Ver imagen 2).

El primer *slider*, al lado derecho de la cuadrícula negra, delimita la densidad de población que se quiere insertar al inicializar el programa. El usuario debe decidir la densidad de población que desea que haya en su ambiente y después dar *click* en el botón Inicializar. La pantalla negra se llenará con una serie de recuadros rojos y azules que representan a los individuos hombres (azul) y mujeres (rojo) que viven en la simulación. Los números que etiquetan a cada individuo representan la fertilidad de cada uno (Ver imagen 3).

Abajo de la cuadrícula hay 3 sliders, uno se llama *Nueva-Fertilidad*, otro *Nueva-Fuerza* y el último *Nueva-Defensa*. En éstos, el usuario debe elegir los valores que se le darán a los individuos modificados genéticamente que se insertarán en la población. Luego, del lado derecho, especificar el número de individuos que se insertarán y al seleccionar el botón *Insertar gen modificado,* los individuos aparecerán en la pantalla con color azul claro (cyan) para los hombres y amarillo para las mujeres (Ver imagen 4).

Para que haya interacciones entre los vecinos de la población se debe seleccionar el botón *Avanzar una vez*. Este botón hace que pase un año (*tick*) en la simulación y que, con eso, se den las interacciones necesarias para la evolución de la población. Los bebés generados por padres no modificados son de color verde y los de padres sí modificados son de color naranja. Las gráficas también dejarán ver la evolución de la simulación. El recuadro blanco ubicado entre gráficas mostrará la cuenta de individuos modificados y no modificados (Ver imagen 5).

Para borrar toda la simulación y empezar desde el principio el usuario debe seleccionar el botón Inicializar en blanco. Este botón limpia los valores guardados, las gráficas y la población.

El botón *Avanzar indefinidamente* ejecuta el botón *Avanzar una vez* infinitamente, o hasta que el usuario lo vuelva a seleccionar.

Conclusiones

Conclusiones generales

En conclusión, este proyecto nos ha ayudado a entender de manera holística lo que conlleva el simular una población de individuos con distintos atributos a través del tiempo, en la que se ha introducido una población de individuos modificados genéticamente.

Una vez concluida la simulación pudimos observar el comportamiento de los agentes y ver como nuestra intervención les afectaba. Si tomamos a la simulación como una representación simplificada de lo que podría suceder cuando se liberen individuos modificados genéticamente en una población normal y estable nos puede ayudar a predecir potenciales escenarios de un fenómeno similar sucediendo sobre individuos reales, cuestión valiosa sabiendo que la ingeniería genética está volviéndose cada vez más común. Los resultados que podemos derivar de experimentar con la simulación son que, bajo nuestro caso acotado, los individuos modificados genéticamente se logran esparcir y dominar a los individuos originales al punto de reemplazarlos. Esto a su vez hace que el promedio global de los atributos de todos los individuos se vea significativamente influenciado por los genes nuevos ya sea para bien o para mal y puede poner en riesgo la estabilidad de la población permanentemente. Esto sucede porque la herencia genética de aquellos individuos que son modificados genéticamente obliga a que se modifiquen sus hijos; es decir, un hijo de un individuo modificado siempre es modificado genéticamente.

Sería necesario complejizar mucho más la simulación para hacer declaraciones más precisas con seguridad y esto vendría con rendimientos decrecientes. Aun así podemos determinar que la inserción de genes modificados en una población altamente interconectada puede tener repercusiones en toda la población y que es necesario tener cuidado al trabajar con estos para limitar el potencial riesgo.

En cuanto al desarrollo de la aplicación, se puede concluir que el programa es amigable con el usuario, hasta el punto de ser de fácil intuición. No se necesita un manual de usuario para entender el funcionamiento básico y es eficiente en la resolución de posibles problemas. Un detalle que se puede mencionar es el trato de la sobrepoblación, pues, después

de varias pruebas, se tuvo que llegar hasta el punto de recortar al 50% de la población viva cada que hay 60% de densidad de población. Esto es necesario porque la cantidad de bebés que nacen cada generación es proporcional al número de posibles emparejamientos entre hombres y mujeres de la población (que cumplan las características especificadas). Entonces, en una población que ya tiene más del 60% de lugares ocupados y que tiene una cantidad de individuos máxima, es muy necesario que se trate la sobrepoblación. Sin embargo, tal vez exista un algoritmo más específico para resolver estos problemas que se puede desarrollar en una siguiente versión del programa.

Por último, el desarrollo de la versión actual del proyecto es muy exitoso, pero puede requerir de algunas mejoras para siguientes versiones. Los objetivos se lograron sin problema y el estudio se pudo hacer con resultados específicos. La plataforma NetLogo hizo posible que el resultado de la aplicación se diera de manera fácil y accesible, pero quizá haberlo desarrollado en otro lenguaje de programación hubiera permitido más flexibilidad en la interfaz.

Anexos

Código

```
breed [women woman]
breed [men man]
breed [babies baby]
globals [menM womenM babiesM]

women-own [fertility age strength defense modif]
men-own [fertility age strength defense modif]
babies-own [inF inS inD modif]

to setup-blank
  clear-all
  set-default-shape women "square"
  set-default-shape men "square"
  set-default-shape babies "circle 2"
  reset-ticks
end

to setup-random
  setup-blank
  let numPat round (Densidad-Inicial * count patches / 100)
  show numPat
  ask n-of numPat patches [
    ifelse random 10 < 5 [
      sprout-women 1 [
        set fertility (random 10) + 1
        set label round fertility
        set strength random 10 + 1
        set defense random 10 + 1
        set color red
        set modif false]
    ][
      sprout-men 1 [
        set fertility (random 10) + 1
        set label round fertility
        set strength random 10 + 1
        set defense random 10 + 1
        set color blue
        set modif false
      ]
    ]
  ]
  reset-ticks
```

```
end


to birth [totalF totalS totalD m]
  ifelse m = false
  [sprout-babies 1 [
    set inF totalF
    set inS totalS
    set inD totalD
    set color lime + 1
    set modif false
  ]
  ] [ sprout-babies 1 [
      set inF totalF
      set inS totalS
      set inD totalD
      set color orange + 2
      set modif true

  ]]
end

to grow
  ask men[
    set age age + 1
    if age > 8
    [set color gray]
  ]

  ask women[
    set age age + 1
    if age > 8
    [set color gray]
  ]

  ask babies [
    let ih (list inF inS inD modif)
    ifelse random 100 < 50 [
      set breed women
      set fertility  (item 0 ih)
      set strength (item 1 ih)
      set defense (item 2 ih)
      set modif (item 3 ih)
      set label round fertility
      set age 1
      ifelse modif = false [ set color red] [set color yellow set
```

```
label-color black]
    ][
      set breed men
      set fertility (item 0 ih)
      set strength (item 1 ih)
      set defense (item 2 ih)
      set modif (item 3 ih)
      set label round fertility
      ifelse modif = false [ set color blue] [set color cyan set
label-color black]
      set age 1
    ]
  ]

end


to insertarMod [cuantos]
  let insertados 0
  while [insertados < cuantos][
    ask n-of 1 patches [
      if not any? turtles-here [
        set insertados insertados + 1
        ifelse random 100 < 50 [
          sprout-women 1[
            set fertility Nueva-Fertilidad
            set strength Nueva-Fuerza
            set defense Nueva-Defensa
            set modif true
            set label fertility
            set color yellow
            set label-color black]
        ] [ sprout-men 1[
            set fertility Nueva-Fertilidad
            set strength Nueva-Fuerza
            set defense Nueva-Defensa
            set modif true
            set label fertility
            set color cyan
            set label-color black
            ]
          ]
        ]
    ]
    ]
end
```

```
to reproduce
  let lst []

  ask men[
    let dad (list fertility strength defense modif)
      ask neighbors [
        ask women-here [
          let mom (list fertility strength defense modif)
          if age < 4 and (item 0 dad + item 0 mom) > 10 [
          ifelse (item 3 dad or item 3 mom)
            [set lst lput (list (max list item 0 dad  item 0 mom) (max
list item 1 dad  item 1 mom) (max list item 2 dad  item 2 mom)
(true))lst] ;; herencia transgénica
          [set lst lput (list (1 / 2 * (item 0 dad + item 0 mom)) (1 / 2
* (item 1 dad + item 1 mom)) (1 / 2 * (item 2 dad + item 2 mom))
(false))lst ] ;; herencia normal
        ]
      ]
    ]
  ]

  foreach lst [
    [?] -> ask n-of 1 patches [
      if not any? men-here and not any? women-here [birth item 0 ? item
1 ? item 2 ? item 3 ?]
    ]
  ]

end

to fight
  ask men[
   if age > 2[
   let s1 strength
   let d1 defense
    ask neighbors [
      ask men-here[
        if age > 2[
            let s2 strength
            let d2 strength
            if s2 < s1  and d2 < d1 and random 100 < 80[
              die
            ]
            if s2 < s1  and d2 > d1 and random 100 < 50[
```

```
            die
          ]
        ]
    ]

    ask women-here[
        if age > 2[
          let s2 strength
          let d2 strength
          if s2 < s1  and d2 < d1 and random 100 < 80[
            die
          ]
          if s2 < s1  and d2 > d1 and random 100 < 50[
            die
          ]
        ]
      ]
    ]
  ]
 ]

ask women[
  if age > 2 [
  let s1 strength
  let d1 defense
  ask neighbors [
       ask men-here[
         if age > 2[
           let s2 strength
           let d2 strength
           if s2 < s1  and d2 < d1 and random 100 < 80[
             die
           ]
           if s2 < s1  and d2 > d1 and random 100 < 50[
             die
           ]
         ]
       ]

       ask women-here[
         if age > 2 [
           let s2 strength
           let d2 strength
           if s2 < s1  and d2 < d1 and random 100 < 80[
             die
           ]
```

```
            if s2 < s1  and d2 > d1 and random 100 < 50[
              die
            ]
          ]
        ]
      ]
    ]
  ]

end

to killPob
  ask men with [color = gray]
    [ die ]
  ask women with [color = gray]
    [ die ]

  show count turtles
  let nueve (count turtles * 0.09)
  ask n-of nueve patches [
    ask turtles-here [die]
  ]

  let indice (count turtles * 100) / count patches
  let cinc round(count turtles * 0.5)
  show indice
  if indice > 60 [
    show cinc
    ask n-of cinc patches[
      ask turtles-here [die]
    ]
    show "se murió el 50 % de la población por sobrepoblación"
  ]

end

to contar
  let cuantosM 0
  let cuantos 0
  clear-output
  set  menM 0
  set womenM 0
  set babiesM 0
  ask men[
    if modif [set menM menM + 1]
  ]
```

```
  ask women[
    if modif [set womenM womenM + 1]
  ]
  ask babies[
    if modif [set babiesM babiesM + 1]
  ]
  set cuantosM menM + womenM + babiesM
  set cuantos (count turtles - cuantosM)
  output-print (word "Indivuduos no modificados genéticamente: "
cuantos)
  output-print (word "Indivuduos modificados genéticamente: " cuantosM)

end

to go
  let lst []

  grow
  fight
  reproduce
  killPob
  contar
  tick
end
```

Imágenes

Imagen 1. Ejemplo de aplicación de escritorio NetLogo

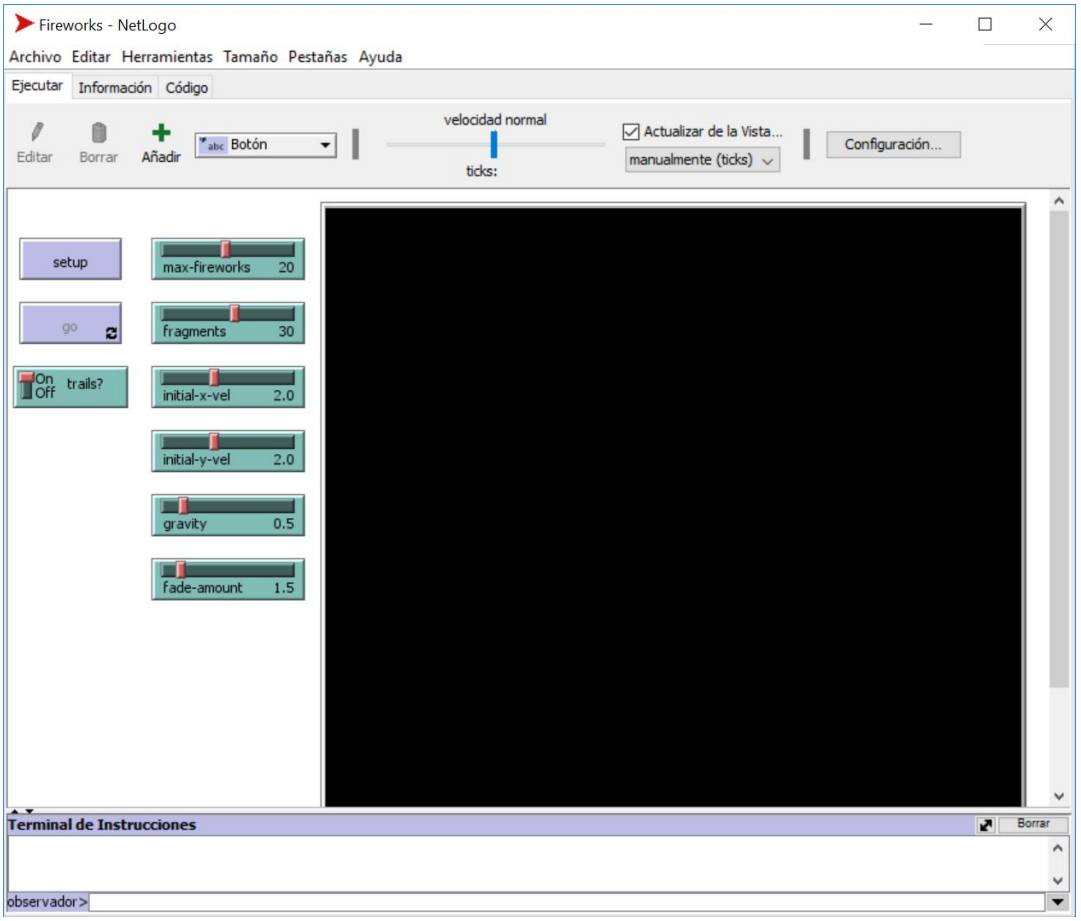


Imagen 2. Ejecución del programa

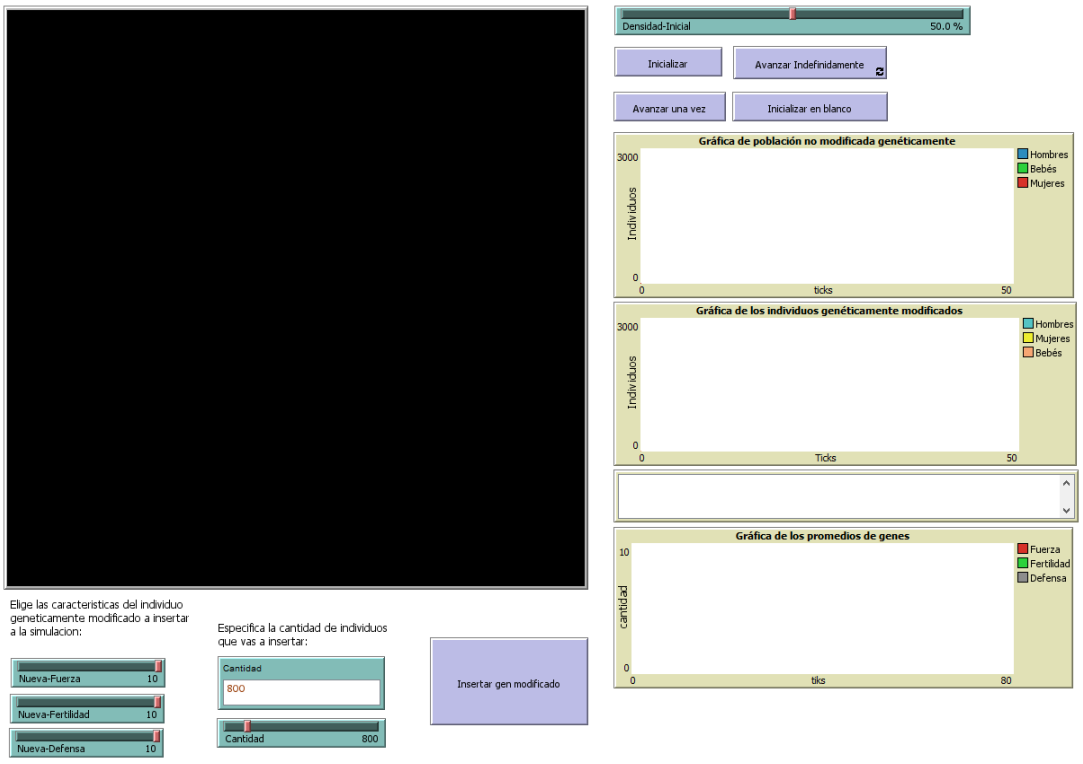


Imagen 3. Click en botón *Inicializar*

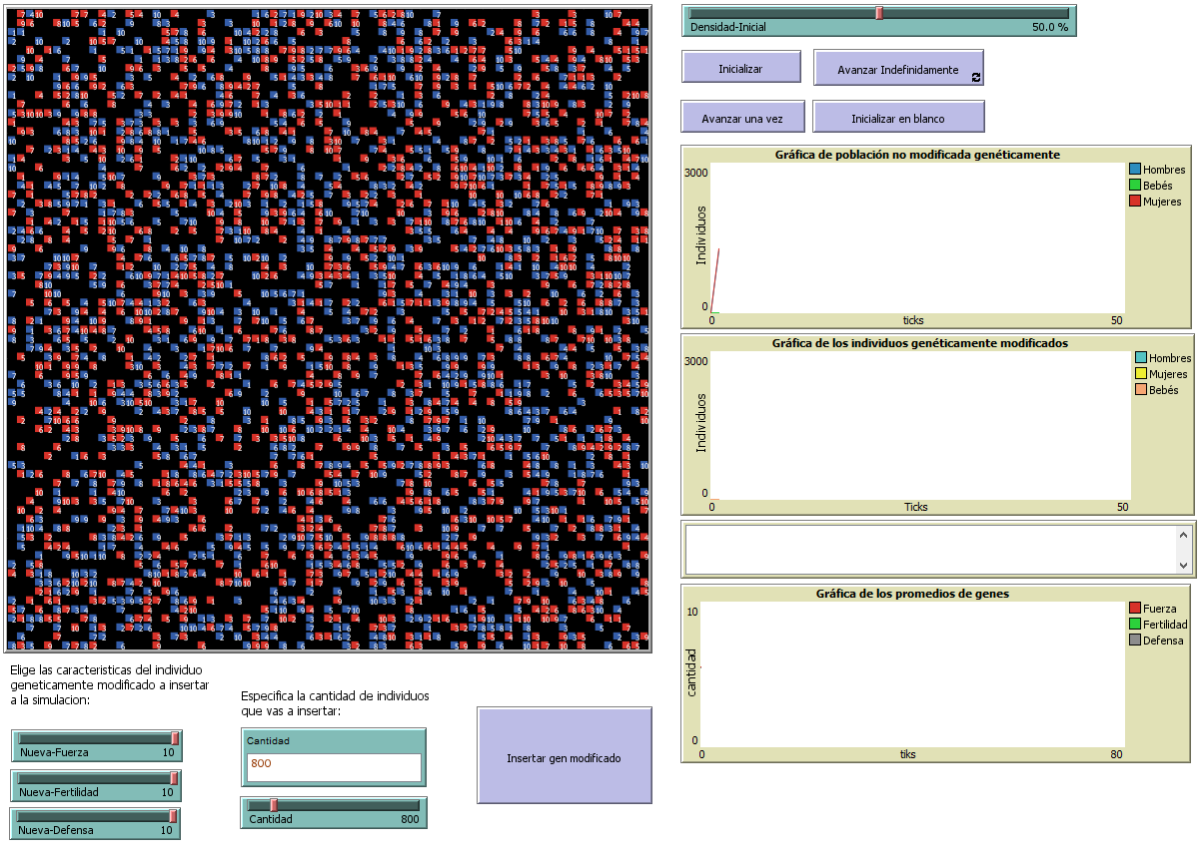


Imagen 4. Click en Insertar gen modificado

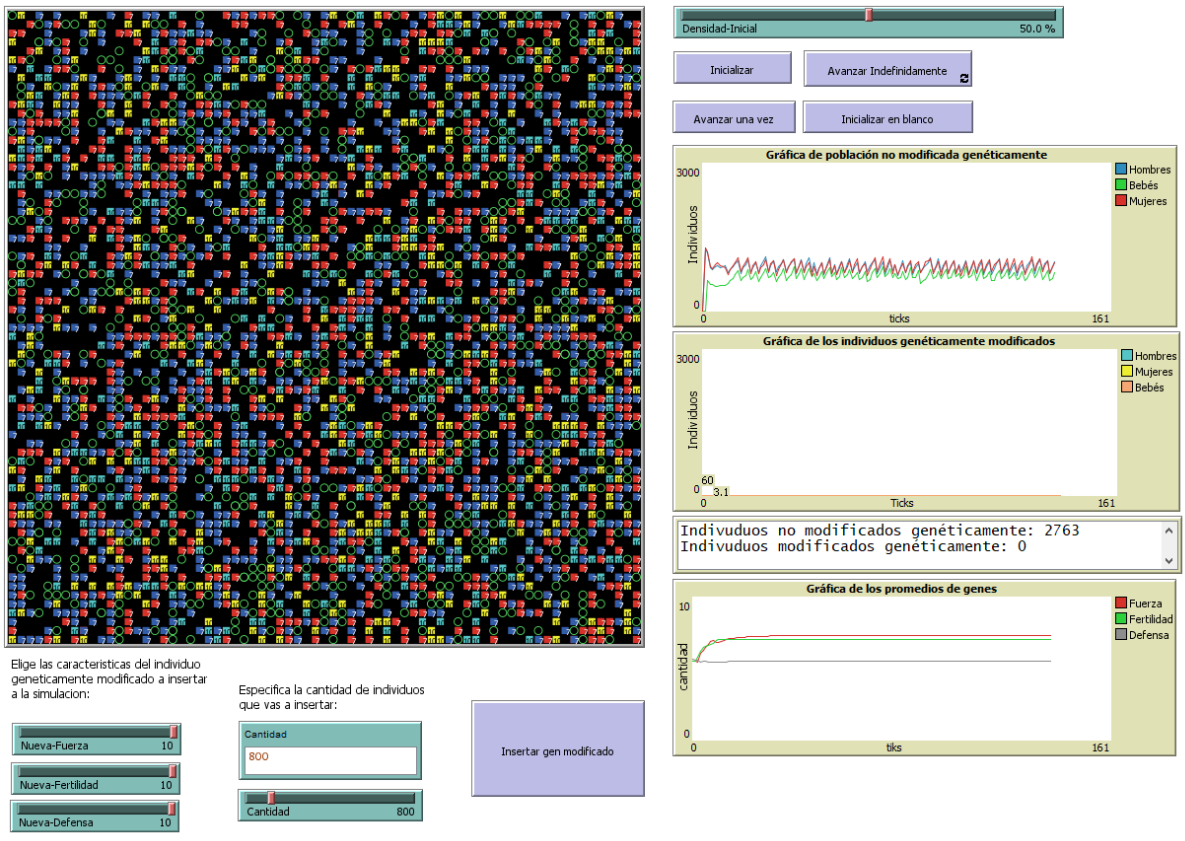


Imagen 5. Individuos genéticamente modificados dominando

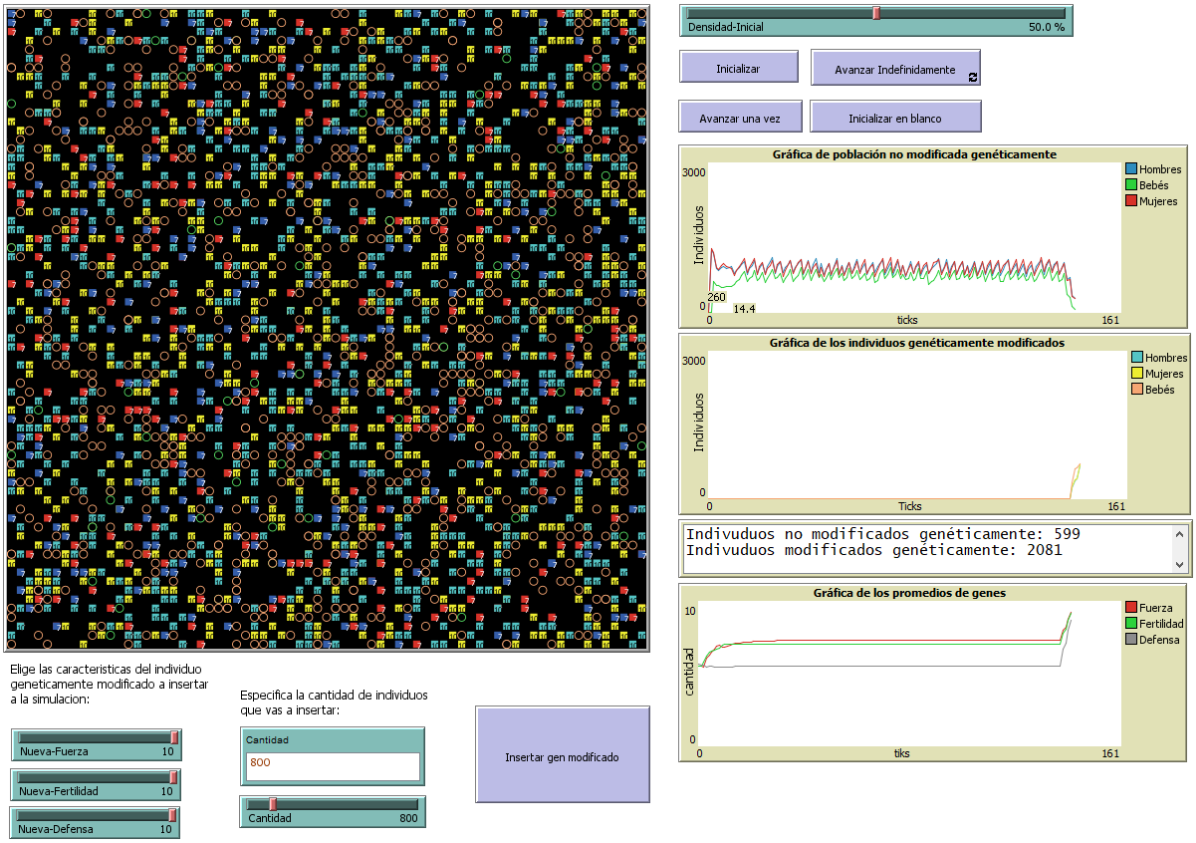
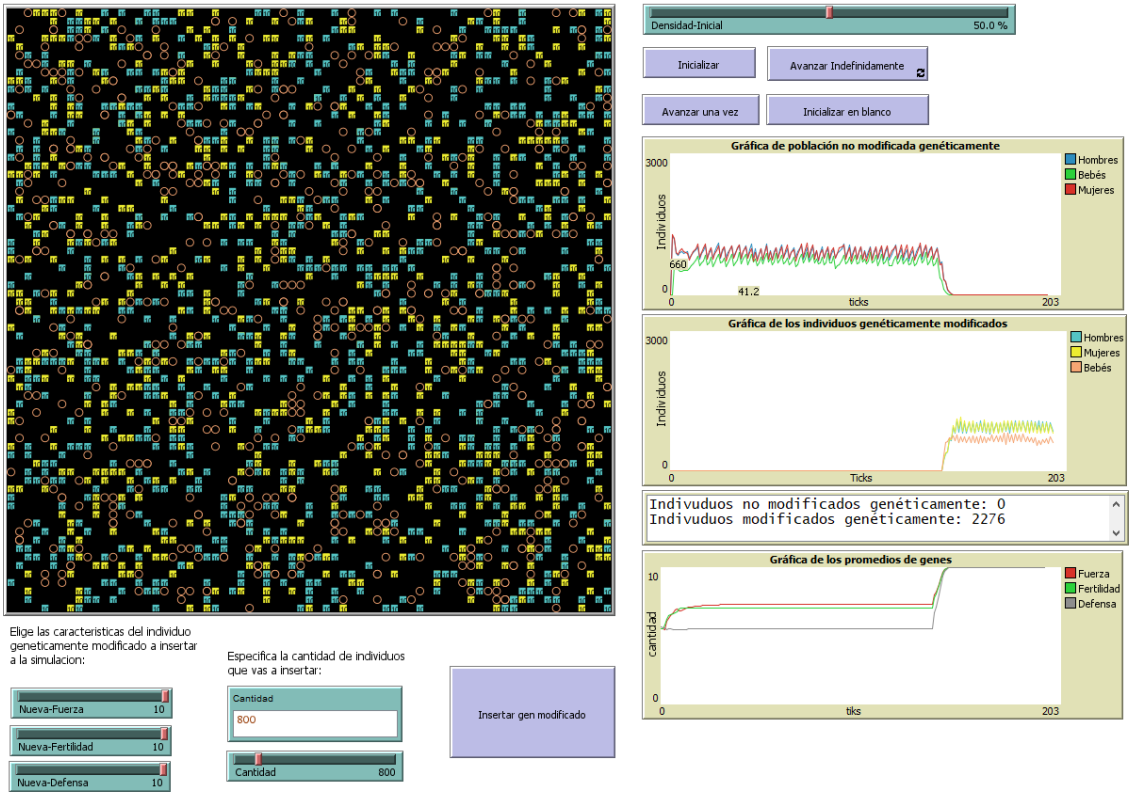


Imagen 6. Individuos genéticamente modificados habiendo reemplazado a los individuos originales y con una población estable



Referencias

[1] Obeso Almeida, L. (n.d.). *Ingeniería genética*. Departamento didáctico: Ciencias Naturales. http://depa.fquim.unam.mx/amyd/archivero/IngenieriaGenetica_13407.pdf.

[2] Roberto I. Ramírez García / José Manuel Segovia Coronel. (n.d.). *Edición genética en humanos, la gran controversia*. Edición genética en humanos, la gran controversia - Revista ¿Cómo ves? - Dirección General de Divulgación de la Ciencia de la UNAM. http://www.comoves.unam.mx/numeros/articulo/246/edicion-genetica-en-humanos-la-gran-controversia.

[3] *Qué es NetLogo* (2018). NetLogo. https://ccl.northwestern.edu/netlogo/resources/Que%20es%20NetLogo.pdf

[4] *Programming Guide*. NetLogo 6.2.0 User Manual: Programming Guide. https://ccl.northwestern.edu/netlogo/docs/programming.html.

[5] *Life Turtle-Based*. Netlogo Models Library. http://ccl.northwestern.edu/netlogo/models/LifeTurtle-Based

[6] *The fantastic combinations of John Conway's new solitaire game "life"*. MATHEMATICAL GAMES . https://web.stanford.edu/class/sts145/Library/life.pdf

[7] *Genetic Engineering*. National Human Genome Research Institute. https://www.genome.gov/genetics-glossary/Genetic-Engineering