Juegos de ordenador

Tiburones y peces libran una guerra ecológica en el planeta Acua-Tor

A. K. Dewdney

En lontananza, mirando en dirección que sólo cabe llamar recreativa, a distancia limitada únicamente por la osadía y pericia de cada cual como programador, nada entre las estrellas el planeta Acua-Tor. Su forma es toroidal, es decir, semejante a una rosquilla, y está enteramente cubierto de agua. Las dos especies dominantes en Acua-Tor son tiburones y peces, así llamados por ser éstas las criaturas terrestres que más se les asemejan. Los tiburones de Acua-Tor se nutren de los peces, que depredan y de los que parece haber siempre abundante provisión.

Este simplista esquema ecológico podría parecer estable y aún soporífico, si no fuera porque las poblaciones de peces y tiburones experimentan violentas fluctuaciones. Muchas veces estuvo en tiempos pasados a punto de ser devorada la población de peces, mientras que en otras ocasiones los tiburones casi llegaron a extinguirse por completo (a pesar de la abundancia de peces). Empero, tanto tiburones como peces sobreviven aún. Buscando descubrir por qué, concebí la idea de un programa que simulase sus actividades tróficas y reproductoras.

No obstante, antes de contemplar esos ritmos ecológicos en la pantalla de visualización [*véanse las figuras 1 y 2*] hube de dedicar no poco tiempo a meditar sobre las reglas y detalles del programa ACUA-TOR. Un día, de sobremesa, me encontré pensando a media voz frente a David Wiseman, el mago especialista en sistemas de mi departamento de la Universidad de Ontario Occidental. Cuando le hube explicado el proyecto, Magi (así le llamamos) sonreía enigmáticamente. Al día siguiente, por la mañana, me invitaba ufano a ver en su oficina un sistema en funcionamiento.

"Fíjate", me dijo, y pulsó una tecla. Un surtido inicialmente aleatorio de peces y tiburones revoloteaba lentamente de uno a otro lugar, caóticamente, al parecer. Algunos tiburones no lograban alimentarse, y desaparecían. Otros generaban progenie, tan voraz como ellos mismos. Unos cuantos peces, lo bastante afortunados para ocupar temporalmente una región libre de tiburones, proliferaron y se multiplicaron hasta formar un notable cardumen. Poco después, algunos tiburones descubrieron el banco de peces, se congregaron en sus bordes y avanzaron un breve trecho, abriéndose paso a dentelladas. Algunos minutos más tarde, el resumen estadístico que presentaba la pantalla de Magi nos daba cuenta de la situación: 578 peces frente a sólo 68 tiburones.

Alguien entró en el despacho de Magi y volvió a salir corriendo. Antes de cinco minutos su despacho estaba atiborrado de gente que jaleaba a los tiburones. Lentamente, una pared de escualos fue cerrándose sobre los desventurados peces. En otra región de la pantalla, un reducido banco de peces medraba, desapercibido. Las exclamaciones subieron de punto cuando el gran cardumen acabó devorado, mientras los tiburones iban uno tras otro muriendo de inanición, arremolinándose y merodeando en busca de presas. Pensé en cambiar las reglas y permitir que los tiburones se devorasen unos a otros, pero comprendí que un tal festín no serviría para prolongar significativamente su existencia, pudiendo, en cambio, comprometer las fases iniciales del otro pequeño banco. Cuando por fin dos tiburones se toparon con él en su merodeo, el ciclo recomenzó.

El programa para el Acua-Tor no es muy largo ni de difícil redacción. Los lectores que dispongan de ordenador personal, aunque tengan reducida experiencia como programadores, encontrarán remuneradora la empresa cuando por fin tengan escritos los códigos, los hayan depurado y los vean funcionar. Antes de cada pasada pueden ajustarse parámetros tales como tiempos de cría, períodos de hambruna y tamaño de las poblaciones iniciales. Lo único que resta entonces es arrellanarse y observar cómo en la mezcolanza desordenada de peces y tiburones comienzan lentamente a crearse pautas ecológicas.

El programa ACUA-TOR lleva incorporado cierto número de reglas sencillas que gobiernan las conductas de peces y tiburones. Las criaturas nadan en un océano rectangular cuadriculado, cuyos lados opuestos están identificados dos a dos. Significa esto, simplemente, que si un pez o tiburón ocupa un punto situado en el borde derecho, y opta por nadar hacia el este (hacia la derecha), reaparecerá en el correspondiente lugar del borde izquierdo. Igual relación es válida para los bordes superior e inferior. El espacio bidimensional así resultante es en realidad un toro, que es la auténtica superficie de ACUA-TOR [*véase la figura 3*]. Quienquiera se decida a redactar su programa ACUA-TOR puede determinar el tamaño más adecuado de la cuadrícula oceánica. Magi, cuyo programa funciona en un ordenador VAX, ha preparado un océano de 80 puntos de ancho y 23 de alto. Mi propia versión de ACUA-TOR, para un IBM PC, se vale de un océano mucho más modesto: 32 por 14.

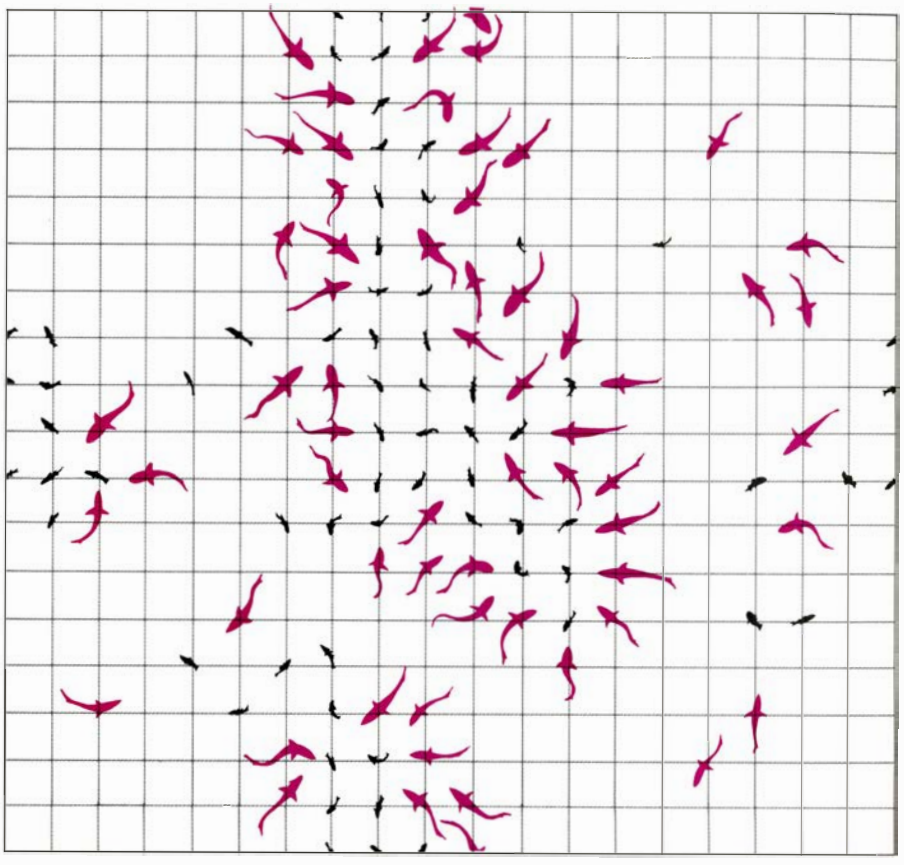
El tiempo transcurre a saltos discretos, que llamaré crononos. En cada cronón, peces y tiburones pueden desplazarse hacia el norte, sur, este y oeste, trasladándose a un punto adyacente, siempre que no esté ocupado ya por otro miembro de su misma especie. La decisión concreta queda a cargo de un generador de números aleatorios. Para los peces, la elección es sencilla: han de tomar al azar alguna de las casillas adyacentes disponibles y trasladarse a ella. Si los cuatro puntos adyacentes están ocupados, el pez permanece inmóvil. Para los tiburones la regla es un poco más complicada, pues la depredación tiene prioridad sobre el mero desplazamiento: de entre los puntos adyacentes ocupados por peces, se se-

lecciona uno al azar y, trasladándose allí, el tiburón devora al pez. Si no hay peces vecinos, el tiburón se mueve exactamente igual que los peces, eludiendo a sus congéneres.

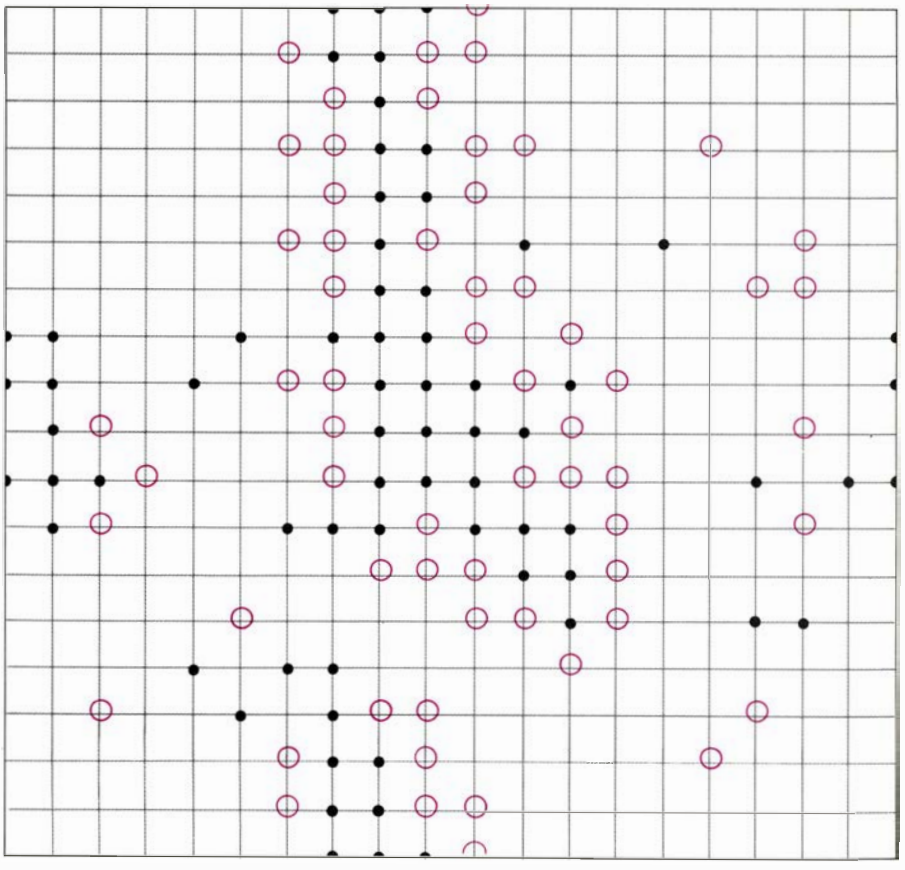
El creador de ACUA-TOR selecciona cinco parámetros, a fin de definir una determinada simulación. Los parámetros *npeces* y *nescualos* representan los números de peces y tiburones al comienzo de cada pasada. El programa distribuye al azar, más o menos uniformemente sobre la superficie del planeta, los números prefijados de peces y tiburones. Los parámetros *pcría* y *tcría* designan los números de cronones que han de sobrevivir peces y tiburones para engendrar una única cría. (Según parece, ambas especies son partenogenéticas.) Finalmente, la variable *ayuno* especifica el número de cronones que puede sobrevivir un tiburón sin hallar comida. Si merodea más de lo estipulado sin echarse nada al coleto, muere, se hunde y desaparece. Durante cada cronón, el programa ACUA-TOR mueve una vez a cada pez y a cada escualo, y muestra en la pantalla el resultado. No son precisas reglas más complicadas que éstas para observar cómo la ecología de ACUA-TOR va a trompicones de una a otra crisis.

Magi y yo hemos observado cierto número de situaciones gobernadas por esos cinco parámetros, donde el océano de ACUA-TOR se sobrepoblaba de peces, con la consiguiente multiplicación de los tiburones, hasta el extremo de devorar a todos los peces y luego morir ellos. En otras ocasiones hemos visto devorar a todos los peces de un gran cardumen. Los tiburones, tras haberse dado el hartazgo, acabaron muriendo, incapaces de dar con otros bancos cercanos. En unas cuantas ocasiones hemos visto sostenerse la relación entre pez y depredador durante dos, y aun tres ciclos, antes de producirse el definitivo hundimiento de la población de escualos. Por otra parte, en los parámetros determinantes de estas situaciones nada nos daba la menor indicación acerca de cuáles serían las características capaces de garantizar un sistema ecológico de duración indefinida. ¿Cómo habían logrado sobrevivir las especies acuatorianas?

Se ha dicho que la biología es destino. Magi y yo estamos tentados de afirmar que la ecología es geometría, al menos por lo que a ACUA-TOR se refiere. La suerte final de cada escenario no parece depender de la aleatoriedad de la distribución inicial de los números de peces y de tiburones. Parece, por el

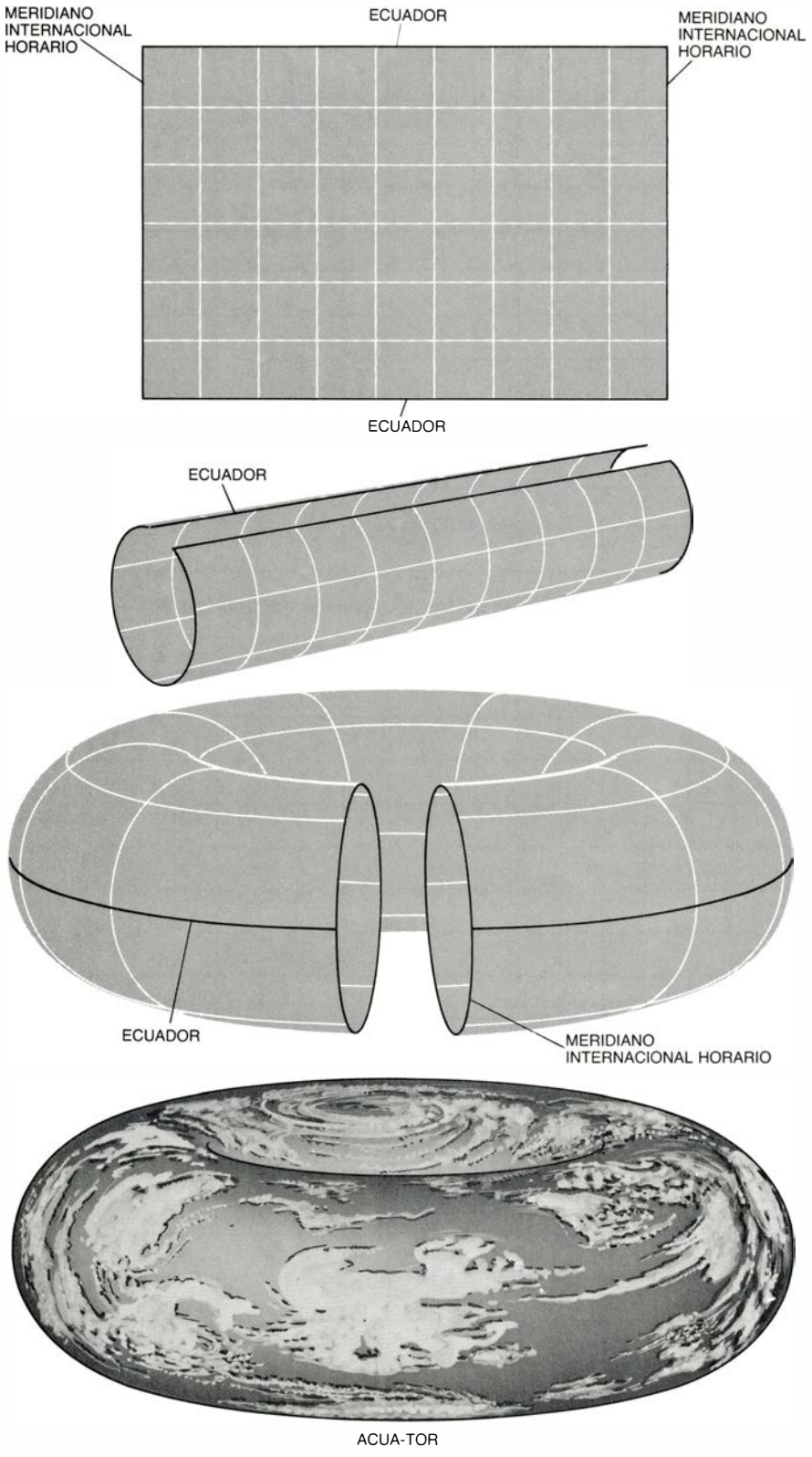


1. Imagen realista de los tiburones devorando peces



2. Una imagen de programación más sencilla. Los círculos representan tiburones y, los puntos, peces

contrario, que la verosimilitud del hundimiento final de la población se ajusta mucho más de cerca a la organización geométrica que se manifiesta en nuestras pantallas: cuanto más altamente organizada y localizada se encuentre cada población, tanto más verosímil es que se produzca una catástrofe en su ecología. Al meditar en este tema nos vimos llevados a conjeturar cómo podríamos elegir los valores de los cinco parámetros, de modo tendente a romper tal estructura. Nos llegó entonces un relámpago de inspiración: si los tiburones llegaran a concentrarse en los bordes de un banco de peces, para romper la estructura geométrica resultante podría ordenarse que los tiburones se reprodujeran menos frecuentemente. Después de todo, la propia congregación resultaba ser consecuencia más de la reproducción que del movimiento.



3. El planeta toroidal Acua-Tor y su representación sobre un mapa plano (o la pantalla de un ordenador)

Antes de que esta hipótesis tomara cuerpo, habíamos elegido tiempos de reproducción sensiblemente iguales para peces y tiburones. Creíamos que tasas reproductoras similares producirían poblaciones equilibradas. Seguramente muchas de las calamidades del mundo tecnológico de hoy sean consecuencia de vagos prejuicios de este mismo tipo. Sea como fuere, situé en mi océano de 32 por 14 doscientos peces y veinte tiburones; y establecí que los peces se reprodujeran cada tres cronones, prohibiendo en cambio que los escualos lo hicieran antes de transcurridos diez. Al cabo de quince minutos de observar el funcionamiento de mi programa (sin duda bastante lento), fuimos recompensados con la casi total recuperación de la inicial caída de población. La organización geométrica, aunque todavía existente, era mera sugerencia. Los cardúmenes de peces adoptaban la forma de conglomerados imprecisos, de bordes desgarrados y, en ciertas zonas de la pantalla, tiburones y peces se entreveraban, más o menos al azar.

Dejé en funcionamiento el programa toda la tarde, alzando ocasionalmente la vista de cuestiones más importantes desde mi mesa de trabajo. El programa siguió funcionando toda la noche, y cuando visité de nuevo mi despacho, tras la clase matinal, encontré que tiburones y peces continuaban en su cíclica existencia. ¡Aquí estaba Acua-Tor!

Hay muchas maneras de llevar a la práctica un programa ACUA-TOR; pero tal vez la más sencilla consista en utilizar cierto número de tablas o matrices bidimensionales. Yo suelo emplear cinco, que llamo PECES, ESCUALOS, MUEVEPECES, MUEVESCUALOS e INANICION. Estas tablas, todas de 32 por 14, llevan la cuenta y control de las edades de peces y tiburones. Concretamente, PECES (I, J) denota la presencia o ausencia de un pez en el punto de coordenadas (I, J). En ausencia de peces, tal posición tiene el valor −1. De haber alguno, contiene un asiento con la edad en cronones del pez ocupante. Un esquema idéntico se utiliza para la tabla ESCUALOS, que se ocupa del control de edades y posiciones de los tiburones. La tabla MUE-

VEPECES contiene en cada casilla un registro de si algún pez ha sido llevado allí durante el ciclo de cómputo en curso. Tal registro evita que el programa mueva dos veces el mismo pez durante un mismo cronón. MUEVESCUALOS lleva a cabo idéntica tarea para los tiburones. La tabla llamada INANICION registra cuál fue la última ocasión en que cada tiburón logró alimentarse. Cuando en la casilla correspondiente no hay tiburón, se le da el valor -1.

La forma más sencilla de presentar lo que en ACUA-TOR acontece es mostrar en la pantalla una línea de caracteres para cada hilera de las tablas. Cuando una posición está en blanco, ello denota que se encuentra vacante el puesto correspondiente. Los puntos (.) representan peces y, los ceros (0), tiburones. Aunque esta presentación parezca muy limitada, resulta sorprendentemente informativa y grata de contemplar.

En la fase inicial de ACUA-TOR, los números requeridos de peces y tiburones se esparcen uniformemente sobre un oceáno de forma tórica. El programa ejecuta entonces cíclicamente los tres segmentos o subprogramas que se describen a renglón seguido. Cada ciclo completo del programa equivale al tiempo de un cronón.

PECES: NATACION Y REPRODUCCION.
Para cada uno de los peces de la tabla PECES, el programa prepara una lista de posiciones desocupadas adyacentes, y traslada el pez hasta una de ellas, elegida al azar. Por tanto, en la antigua posición de PECES habrá de inscribirse un -1 y, en la nueva, la edad actual del pez. La tabla MUEVEPECES ha de actualizarse igualmente según acabamos de explicar. Cuando la edad del pez es igual a *pcría*, el programa sitúa otro pez en la posición primitiva, y les asigna a ambos la edad 0. Como antes, MUEVEPECES toma nota del nuevo pez. Si todas las posiciones adyacentes estuvieran ocupadas, el pez no se movería ni procrearía.

TIBURONES: CAZA Y REPRODUCCION.
Para cada uno de los tiburones de la tabla correspondiente, el programa prepara una lista de posiciones contiguas ocupadas por peces (si las hay). El tiburón elige al azar una de ellas, se dirige allí y devora el pez. Así pues, el programa deberá actualizar las tablas de ESCUALOS y MUEVESCUALOS, como ya hizo con PECES y MUEVEPECES, y además asignar el valor -1 a la correspondiente posición de la tabla PECES. Asimismo, ha de poner a 0 esa misma posición en la tabla INANICION. Cuando no hay peces adyacentes, los tiburones se mueven igual que los peces. Si la edad de un tiburón alcanza cuando menos la de *tcría*, se engendra un nuevo tiburón, como ocurre con los peces.

PRESENTACION:
El programa inspecciona las matrices de PECES y ESCUALOS; presenta un punto para cada pez y un 0 para cada tiburón. La presentación puede ser simultánea, o puede hacerse en dos partes: una después de mover los peces y, otra, después de la traslación de los tiburones.

Para poblar el océano inicial, el programador construye un bucle con el que genera *npeces* parejas de números aleatorios. Tales números deberán ajustarse a las dimensiones horizontal y vertical del océano que elija. En cada una de las posiciones aleatorias así seleccionadas, el programa sitúa un pez en la tabla PECES, asignándole también al azar una edad comprendida entre 0 y *pcría*. La distribución de tiburones sigue un método análogo. En ambos casos, es preciso comprobar antes si la posición elegida está ocupada ya. Esta asignación aleatoria de edades a peces y a tiburones tiene por efecto que su reproducción se efectúe también al azar, de modo natural. De no ser por esta precaución, veríamos cómo peces y tiburones duplicaban súbitamente su número, lo que produciría una sensación tan desconcertante como poco natural.

Tal vez haya programadores noveles para quienes la descripción anterior resulte demasiado general y no se hagan idea clara de cómo redactar un programa ACUA-TOR. Tales programadores podrían comenzar preparando lo que se conoce por programa del borracho, que al caminar vacila y se tambalea. Tal programa podría consistir en un único bucle (del tipo "haz-mientras", por ejemplo), que consta de siete instrucciones. Las hemos escrito aquí en lenguaje algorítmico, que no prejuzga ningún lenguaje de programación específico. Las asignaciones de valor están indicadas por flechas que apuntan hacia la izquierda; las variables X e Y son las coordenadas del vacilante beodo, y van modificándose de acuerdo con el entero aleatorio que le esté asignado a la variable *dirección*. Según este entero sea 0, 1, 2 o 3, el borracho (un punto de la pantalla) se mueve al norte, al este, al sur, o al oeste.

dirección $\leftarrow$ parte entera
  de (*azar* $\times$ 4)
si *dirección* = 0 entonces $X \leftarrow X + 1$
si *dirección* = 1 entonces $X \leftarrow X - 1$
si *dirección* = 2 entonces $Y \leftarrow Y + 1$
si *dirección* = 3 entonces $Y \leftarrow Y - 1$
presentar (X, Y)

Si nuestro generador particular de números aleatorios engendra un número decimal, llamado *azar*, comprendido entre 0 y 1, este algoritmo ajusta el margen de valores aleatorios desde 0 hasta 0,999. La parte entera del número resultante será entonces 0, 1, 2 o 3.

No insinúo que observar por la pantalla de un monitor los minúsculos desplazamientos de un punto luminoso sea lo mismo que el drama ecológico que protagonizan tiburones y peces, pero la redacción de este programa sí nos facilita la comprensión de cómo podrían construirse las secciones de ACUA-TOR.

A los programadores expertos que lean esta sección se les ocurrirán sin duda otros métodos para redactar el programa ACUA-TOR. Puede reducirse notablemente el volumen de datos a procesar utilizando listas encadenadas para el seguimiento de peces y tiburones. Con tal estructura de datos, el tiempo requerido para cada ciclo de cómputo es proporcional al número de peces y tiburones presentes, y no al tamaño del océano.

ACUA-TOR puede proporcionarnos cierta comprensión de las poblaciones animales de aquí, en la Tierra. Sabemos que las poblaciones pequeñas afrontan elevada probabilidad de extinción, y que, aún en el caso de que ni presa ni predador lleguen a extinguirse, es casi seguro que sufrirán fluctuaciones en su número. En ecosistemas sencillos, de tipo depredador-presa, sus poblaciones experimentan dos ciclos superpuestos de máximos y mínimos de población. Los tamaños de las poblaciones de liebres mímicas y de linces registrados desde 1847 hasta 1903 en la región subártica canadiense se ajustan a esta pauta [*véase la figura 4*]. Tales cifras dan el número de ejemplares de cada especie atrapados durante el correspondiene período de un año. Es de presumir que los valores sean proporcionales a los tamaños auténticos de las poblaciones. Si lo fueran, los ciclos se explicarían fácilmente como consecuencia de que los linces medran y pro-

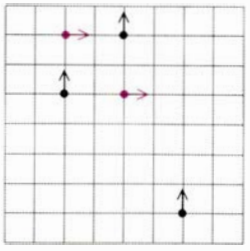
liferan a expensas de una población de liebres en crecimiento, pero cuyo declive comienza conforme aumenta el número de los linces. Pronto empieza a escasear la comida de los linces, y éstos van muriendo de inanición, o se reproducen menos, o ambas cosas. Al reducirse el número de linces, las liebres comienzan, nuevamente, a multiplicarse.

Para compararlas con este gráfico tenemos un juego de curvas lisas que representan una solución de las ecuaciones de Lotke-Volterra, formuladas por primera vez por V. Volterra, un matemático italiano, en 1931. En ellas se supone lo que podríamos llamar un depredador "continuo", permanentemente en busca de una presa, también continua. Las soluciones de estas ecuaciones exhiben una variación cíclica que, a primera vista, parecen reflejar los datos empíricos correspondientes a liebres y linces. No obstante, los biólogos no están de acuerdo con que las cifras de liebres y linces se expliquen con tan sencillo razonamiento. Pues, para empezar, intervienen al menos otros dos depredadores de liebres, a saber, los microbios y el hombre.

Por otra parte, la recopilación de estadísticas sobre tiburones y peces de Acua-Tor tiene pleno sentido, y así lo hemos hecho Magi y yo. Nuestros recientes gráficos de las poblaciones de peces y tiburones tienden a parecerse más a la de linces y liebres que las soluciones de la ecuación de Lotke-Volterra. Seguimos, no obstante, intrigados por la inestabilidad que muestran a largo plazo ciertas combinaciones de parámetros. Tal vez algún lector, trabajando con un programa ACUA-TOR propio, proporcione alguna luz. ¿Existe algún tipo de regla general por la que podamos predecir, dada una cierta combinación de parámetros, si el sistema ecológico resultante será estable? ¿En qué medida se ajustan las fluctuaciones cíclicas a lo previsto en las ecuaciones de Lotke-Volterra?

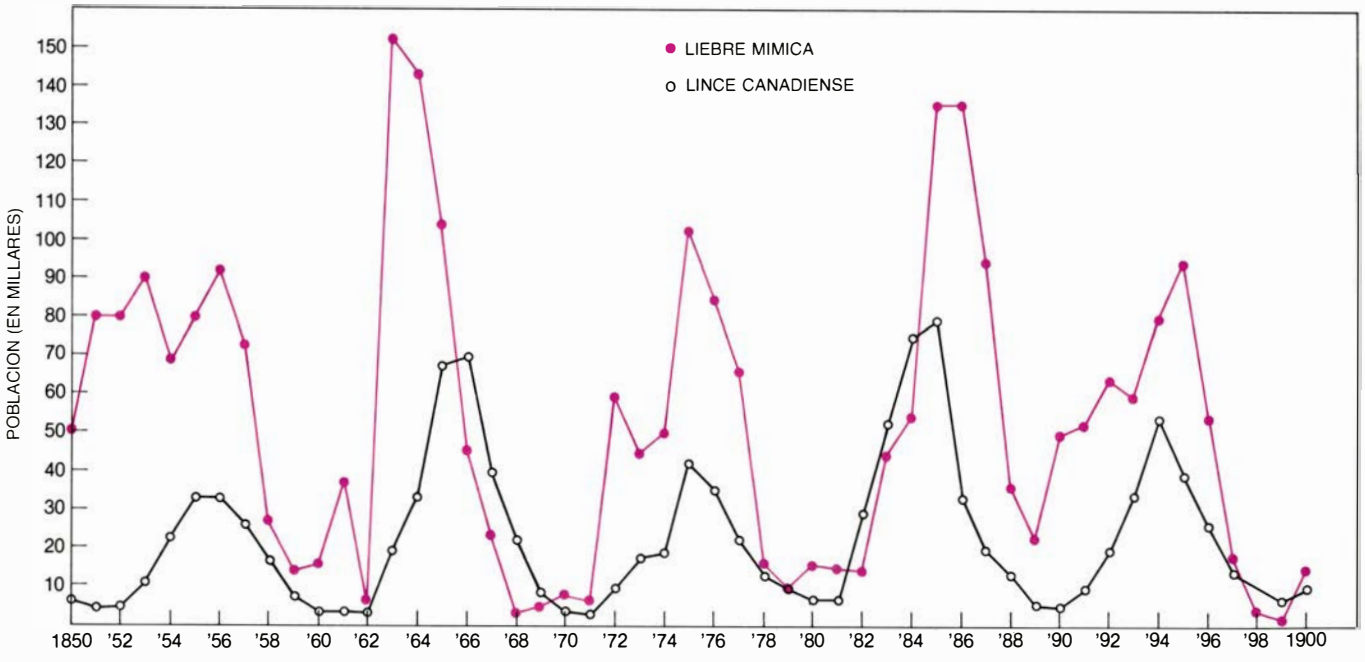
El océano de Acua-Tor es toroidal por una razón muy sencilla: la redacción del programa se simplifica mucho si el océano carece de contornos o riberas. Supongamos que el océano haya de tener 32 unidades de anchura; es cosa sencilla utilizar números módulo 32 para las abscisas (coordenadas horizontales) de peces y tiburones. Si alguno de éstos ocupara en un cronón dado un punto de abscisa 31, apareciendo por tanto en el borde derecho de la pantalla, podría perfectamente tener en el cronón siguiente abscisa 32 (= 0) y presentarse en el borde izquierdo. El mismo sistema se usa en sentido vertical, para las ordenadas.

El océano toroidal de Acua-Tor da pie a ciertos efectos muy extraños, como ponen de manifiesto los siguientes problemas. En el primero de éstos se saca partido de un desliz cometido en una de las primeras versiones de mi programa ACUA-TOR. A causa de este desliz, en cada cronón los peces nadaban una unidad hacia el norte y, los tiburones, una hacia el este. Así pues, la única forma de que un tiburón pudiera comerse un pez era encontrarse ocupando la misma casilla que su presa. Dado el océano que se muestra aquí, ¿cuántos fueron los peces que nunca pudieron comerse los tiburones?

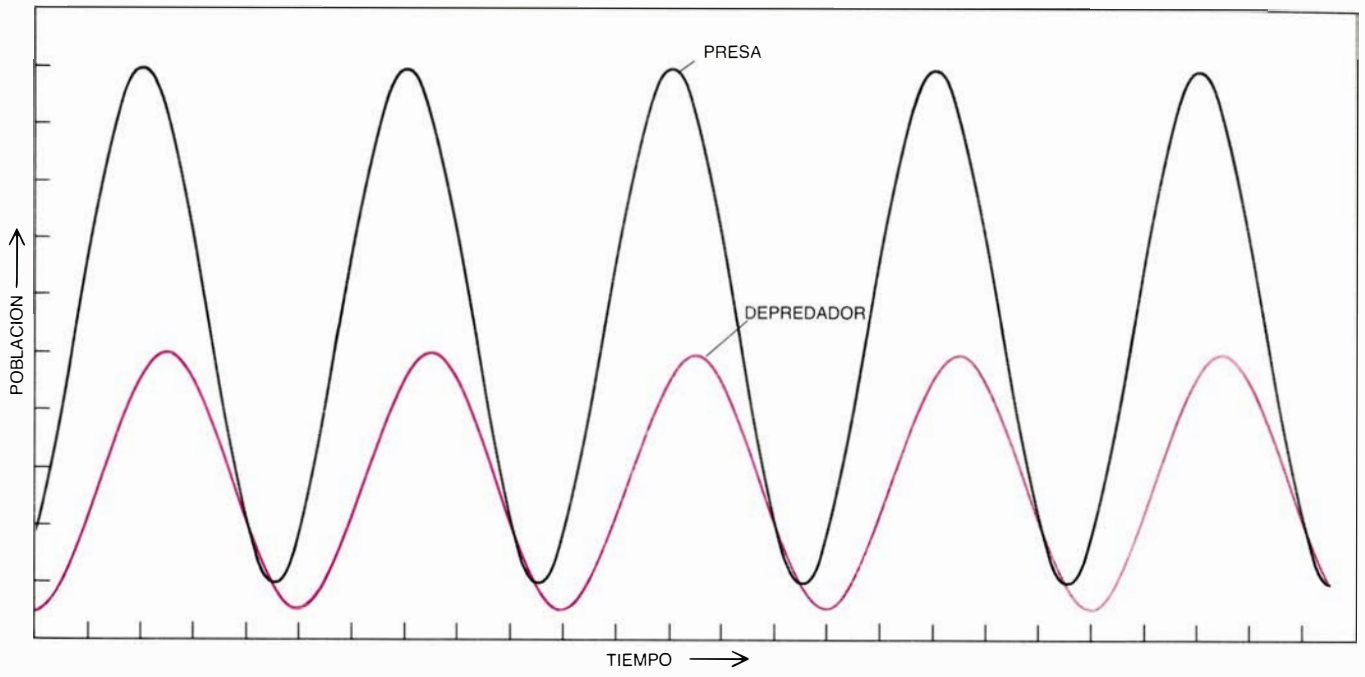


En otro de los problemas se supone que peces y tiburones son inteligentes. Imaginemos que cada pez y cada tiburón va por turno trasladándose hasta cualquiera de sus cuatro puntos vecinos. Resulta que un único pez, si es lo suficientemente inteligente, puede siempre escapar de un único tiburón, por muy inteligente que sea el depredador. En el océano tórico de Acua-Tor pudiera suceder que dos tiburones, acosando a un único pez, lograsen un resultado diferente. Si dotamos a cada criatura de tanta inteligencia como queramos, permitiendo incluso que los tiburones cacen cooperativamente, ¿podremos descubrir una vía de escape para el pez? El resultado no depende de las dimensiones del océano.

El artículo sobre perceptrones ("Juegos de ordenador", noviembre de 1984) les sugirió aplicaciones a unos



4. Número de linces y liebres (en unidades de millar) capturados anualmente para la Hudson's Bay Company, desde 1850 hasta 1900



5. Una relación teórica presa-depredador: una solución de las ecuaciones de Lotke-Volterra

lectores, mientras a otros los aguijó a emprender investigaciones sobre la cuestión por cuenta propia. Ed Manning, de Stratford, Connecticut, construyó hace diez años un perceptrón, diseñado para convertir imágenes reales en los cuadrados digitizados de la retina de un perceptrón. Manning fue una de las pocas personas que detectaron un error en el perceptrón de ventana capaz de reconocer uno o varios rectángulos, que se mostró al pie de la página 152 del número de noviembre: las cuatro últimas subpautas deberían ser cada una mitad de color y mitad blanca. Manning se preguntaba si el error se había provocado de manera "deliberada, para sondear el nivel de lectura". Me tienta decir que sí.

Gary D. Stormo, un investigador del departamento de biología molecular, celular y experimental de la Universidad de Colorado en Boulder, se ha valido de la noción de perceptrón para lograr el reconocimiento automático de pautas. En concreto, Stormo ha construido una función de ponderación al objeto de reconocer puntos de ligadura o enlace en secuencias de nucleótidos del ARN mensajero, y se vale del teorema de convergencia de los perceptrones para llevar el rendimiento del suyo al nivel óptimo. Los resultados han sido muy alentadores: el perceptrón reconoce con "éxito sustancial" los puntos de enlace.

Todo perceptrón de ventana que incluya en su lista de subpautas una ventana totalmente en blanco o totalmente en negro es un "buen" perceptrón. Lowell Hill, de Venice, California, se fijó en ello y se preguntó si una retina enteramente en blanco o en negro constituye una imagen legítima. La respuesta dependerá del tipo de motivos que se quiera reconocer. En el caso de percepción de rectángulos, parece razonable admitir que una retina negra sea, sencillamente, un gran rectángulo.

En el curso de una incursión sumamente fructífera en el mini-proyecto de investigación sugerido por mí, Constantine Roussos, del Lynchburg College Computer Center, de Lynchburg, Virginia, optó por excluir los perceptrones de ventana con subpautas enteramente en blanco o enteramente en negro. Entre sus logros se cuenta una caracterización de los buenos perceptrones (aquellos que reconocen cuando menos un patrón). Su caracterización se vale de relaciones de traslación entre las subpautas de ventanas que figuran en la lista del perceptrón. Al desplazar una unidad según cualquiera de las cuatro direcciones principales a cada una de esas subpautas, la figura resultante tiene que ser otra de las subpautas de la lista.

Roussos profundizó entonces en los perceptrones de ventana minimales, o sea, aquellos cuya lista de subpautas no pueden ulteriormente reducirse sin destruir la bondad del perceptrón. Tales perceptrones son los ladrillos con los que construir el conjunto de los buenos perceptrones. Roussos escribió un programa de ordenador que descubrió todos los perceptrones de ventana minimales que tenían tamaños de lista de órdenes 2 a 5. No existen perceptrones de ventana de orden 6 que sean minimales. Roussos lanza entonces un desafío, volviendo del revés la tarea por mí propuesta en aquel artículo. Yo pedía que los lectores hallasen una pauta que fuese reconocida por un perceptrón dado; Roussos, en cambio, sugiere hallar perceptrones capaces de reconocer una pauta determinada, dada de antemano.

John M. Evans, de Hartford, Connecticut, achaca los fallos de los perceptrones a las limitaciones inherentes a una jerarquía de sólo dos niveles, compuesta por los demonios locales y un único demonio jefe, al que informan. Introduciendo una especie de gerencia demoníaca intermedia, Evans logra vencer las limitaciones sobre reconocimiento de la conexión que Minsky y Papert descubrieron para los perceptrones ordinarios. Los propios demonios de nivel bajo constituyen una especie de retina, cuyos blancos y negros corresponden a si ciertos demonios informan o no. Un segundo estrato de demonios vigila la pauta creada por los demonios del nivel bajo e informa al demonio-jefe de la presencia de subpautas. Un dispositivo de tres niveles logra distinguir cuáles de los cuatro patrones de prueba son conexos y cuáles no lo son.