TRABAJO FIN DE GRADO

Doble Grado en Ingeniería Informática y Matemáticas

Juegos de vida: simulación y caracterización

Autor
Daniel Jiménez López

Tutores
Antonio Miguel Lallena Rojo
Juan Antonio López Villanueva

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, 15 de junio de 2019

Juegos de vida: simulación y caracterización

Daniel Jiménez López

Palabras clave:

Resumen

Games of Life: simulation and characterization

Daniel Jiménez López

Keywords:

Abstract

Índice general

1 Introducción

El autómata celular fue inventado por von Neumann y Ulam en 1950 para estudiar el problema de construir máquinas artificiales que se reproduzcan a sí mismas [1]. Con el fin de imitar el comportamiento de los seres vivos, el diseño de dichas máquinas incluye el espacio en el que se desarrollan, representado por una malla rectangular en la que en los nodos se sitúan células y éstas evolucionan simultáneamente de acuerdo a un conjunto de reglas simples que dirigen la *física* de su pequeño universo abstracto.

El juego de vida es un autómata celular propuesto por Conway en 1970 y popularizado por Gardner en el mismo año [2]. Consiste en la evolución de una disposición inicial de nodos ocupados con células en una malla rectangular infinita. Dicha evolución viene regida por un conjunto de reglas que se aplican simultáneamente a todas las células en cada iteración.

Uno de los motivos que atrajo la atención de científicos de diversos campos hacia el juego de vida fue observar cómo patrones complejos surgen de la aplicación de un conjunto muy simple y reducido de reglas. De esta manera comenzaron a analizarse configuraciones iniciales que daban lugar a comportamientos interesantes, tales como *glider* (Figura 1.1a) que se desplaza sobre la malla, *blinker* (Figura 1.1c) que retorna a su configuración inicial tras un número finito de iteraciones o *block* (Figura 1.1b) que no ve alterado su forma en cada iteración.

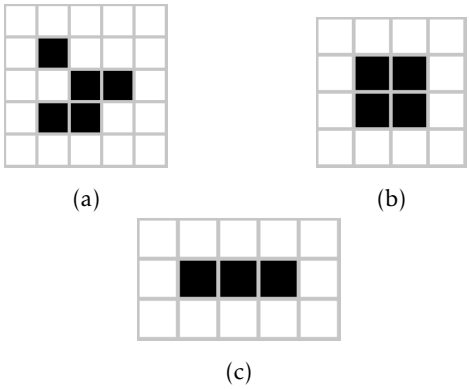


Figura 1.1: Algunas configuraciones iniciales básicas.

Al intentar realizar simulaciones de juegos de vida, los ordenadores se erigieron como la herramienta principal para llevarlas a cabo, siendo necesario afrontar el problema de representar una malla rectangular infinita en un ordenador con memoria finita. Una inteligente solución es alterar las características topológicas de la malla rectangular, identificando los bordes opuestos para obtener superficies topológicamente equivalentes a las de una botella de Klein, una esfera o un toro. En particular, esta última resultó atraer gran interés, pues se encontraron evidencias de que reduce los efectos asociados a la finitud de la malla [3, 4]. Cabe también descatar el estudio de la alteración de las cualidades geométricas de la malla, tales como el uso de figuras geométricas como el triángulo, el hexágono [5], teselaciones de Penrose [6] o el empleo del espacio geométrico hiperbólico [7]. Finalmente, existen implementaciones en las cuales no se almacena la malla en la memoria, si no que se almacena la posición de cada nodo ocupado respecto a un origen de coordenadas [8].

El juego de vida de Conway también muestra interesantes características en el campo de la teoría de la computación, ya que pertenece a la clase IV de Wolfram, esto es, su evolución lleva a estructuras aisladas que muestran un comportamiento complejo [9, 10]. Se ha demostrado que tiene la capacidad de cómputo de una máquina de Turing universal. Por tanto existe una disposición inicial de nodos ocupados que simula una máquina de Turing, la cual fue extendida a una máquina universal de Turing [11, 12]. Como muestra de dicha capacidad de computación, en un esfuerzo colectivo se ha implementado sobre el juego de vida un ordenador con su propios lenguajes de bajo y alto nivel, y se ha programado el conocido juego Tetris [13, 14].

Si los autómatas celulares son un modelo que representa organismos vivos se podría pensar que la hipótesis de actualización simultánea es cuestionable ya que en la naturaleza no se propaga la información de manera instantá-

nea y mucho menos de manera perfecta. Es ese aspecto, que está ligado a la robustez del modelo frente a perturbaciones en su evolución, el que pretendemos analizar en este trabajo. Un modelo será robusto si pequeños cambios en su evolución se traducen en pequeñas perturbaciones del comportamiento global del sistema, mientras que si esta pequeña modificación produce un cambio cualitativo en la dinámica, el sistema será poco robusto o, simplemente, se tratará de un sistema muy sensible a las variaciones que puedan ocurrir en sus elementos. En algunos contextos, los cambios cualitativos a los que nos referimos aquí se conocen como transición de fase. En la bibliografía la introducción de estas perturbaciones se lleva a cabo a través de la asincronicidad en la aplicación de las reglas de evolución [15]. Se pueden considerar las siguientes opciones:

- **Evolución totalmente asíncrona**. En cada iteración, las reglas de evolución se aplican solamente a un individuo escogido del conjunto de nodos, el criterio de elección puede ser o no ser completamente aleatorio.
- **Evolución α-asíncrona**. En cada iteración, cada nodo tiene probabilidad α de que se le apliquen las reglas de evolución y probabilidad $1 - \alpha$ de mantener su estado.

Estos esquemas de evolución también se conocen como **evolución guiada por pasos** y **evolución guiada por tiempo**, respectivamente [16]. Los primeros en estudiar los efectos de la evolución asíncrona frente a la evolución síncrona en el juego de vida fueron *Blok y Bergersen* [17]: aplicando un esquema de evolución α-asíncrona demostraron la existencia de una transición de fase de un comportamiento *estático*, donde el sistema termina alcanzando alguna situación completamente estable, a un comportamiento *vívido* y, por tanto, inestable. Posteriormente se estudió cómo afectaban las variaciones en la topología de la malla a la transición de fase, concluyendo que su aparición depende fuertemente de la regularidad de la malla rectangular [18]. Debido a que no existe una definición globalmente aceptada de autómata celular que agrupe los dos esquemas de actualización asíncrona anteriores, asumimos la definición de autómata celular m-asíncrono dada en [19]. Este autómata implementa un tipo de *oráculo* que selecciona los nodos de la malla que evolucionan en cada iteración.

Hasta donde podemos saber solo se ha estudiado el comportamiento en situaciones de α-asincronicidad de configuraciones iniciales aleatorias, también conocidas como *sopas*, con una densidad prefijada, obteniendo resultados que se comparan con las características conocidas del juego de vida síncrono. En

este trabajo queremos caracterizar la manera en la que configuraciones iniciales bien conocidas y estudiadas, alteran su comportamiento en situaciones de α-asincronicidad. Utilizaremos las técnicas Monte Calo para medir diferentes variables tales como el crecimiento de la población, el número de cúmulos de nodos ocupados y la actividad de cambio.

El capítulo 2 está dedicado a una descripción de los aspectos esenciales de nuestro trabajo. A continuación el capítulo 3 se proporciona una visión más formal de los conceptos expuestos en el capítulo anterior. En el capítulo 4 se documentan los resultados obtenidos y se discuten los aspectos más relevantes de los mismos. Por último en el capítulo 5 se establecen las conclusiones de nuestro trabajo.

2 Aproximación conceptual

En este capítulo daremos una descripción conceptual del universo del juego de vida de Conway junto con la del juego de vida de Conway α-asíncrono. En este contexto se describe en qué han consistido las simulaciones y sobre qué configuraciones iniciales se han realizado.

2.1. Juego de vida de Conway

Una configuración del juego de vida de Conway es la disposición sobre la malla de un conjunto de nodos ocupados. Por tanto una configuración inicial hace referencia a la disposición inicial de nodos ocupados sobre las que aún no se han aplicado las reglas de evolución. De esta manera cada nodo de la malla tiene dos estados posibles: vacío (0) o ocupado (1).

Las reglas de evolución que se aplican en cada iteración simultáneamente a todos los nodos de la malla son las siguientes:

- Un nodo ocupado se mantiene así si en su vecindario tiene solamente dos o tres nodos ocupados, en otro caso el nodo se vacía.
- Un nodo vacío se ocupa cuando en su vecindario hay exactamente tres nodos, en otro caso mantiene su estado vacío.

Usualmente se consideran como pertenecientes al vecindario los nodos adyacentes en las direcciones horizontal, vertical y diagonales.

En lo que hemos interpretado como un sentido *biológico*, las reglas se pueden describir también interpretando los nodos ocupados como células vivas y la configuración inicial como una población de células:

- Una célula puede *morir de soledad*, es decir, tiene solamente una célula en su vecindario o por *superpoblación*, esto es, tiene cuatro o más células en su vecindario.
- En un nodo vacío *nace* una célula si en su vecindario hay exactamente tres células vivas.

La elección de las reglas de evolución parecería *a priori* aleatoria, sin embargo, Conway las escogió aplicando las siguientes pautas [20]:

- No debe existir una disposición inicial de nodos ocupados para la cual haya una *prueba simple* de que la población crezca sin límite. Esto es, no debe de ser posible predecir fácilmente la evolución de una configuración inicial.
- Debe haber disposiciones iniciales de nodos ocupados que aparentemente crezcan sin límite.
- Debe haber disposiciones iniciales de nodos ocupados *sencillas* que crezcan y cambien durante un periodo relativamente largo, llegando a tres posibles finales: desaparecer completamente ya sea debido a superpoblación o a dispersión, estabilizarse en una configuración que se mantenga constante o entrar en un ciclo sin fin de oscilación.

Sin embargo las reglas que proporcionó Conway no son las únicas que muestran evoluciones interesantes. La siguiente notación nos es útil para expresar distintas reglas de evolución abreviadamente. La reglas de evolución vienen dadas en la forma Bx/Sy donde y define el número de nodos ocupados en el vecindario para que un nodo ocupado se mantenga y x el número de nodos ocupados en el vecindario para que un nodo vacío se ocupe. Por ejemplo, para el juego de vida es B3/S23. Destacamos la regla B1357/S1357 conocida como *Edward Fredkin's replicating automaton*, en la cual cada configuración es eventualmente reemplazada por múltiples copias de sí misma, la regla B3/S12345 conocida como *Maze*, genera configuraciones similares a laberintos, una variación muy curiosa es que si añadimos el número 7 a la parte B es posible observar como un nodo ocupado recorre el laberinto y la regla B3/S012345678 conocida como *Life without Death*, en la cual los nodos ocupados nunca se vacían, se caracteriza por un crecimiento caótico y la aparición de configuraciones similares a escaleras que pueden ser usadas para simular circuitos booleanos [21].

Representación interna y actualización

Como se comentaba en la introducción, plantearse la simulación del juego de vida implica afrontar el problema de representar una malla infinita de dos dimensiones en la memoria finita de un ordenador. Aunque la cantidad de memoria y velocidad de acceso a la misma ha mejorado significativamente con el paso del tiempo, perseguimos una representación que cumpla las siguientes dos características:

- Una simulación de una configuración inicial del juego de vida tiene que finalizar en un tiempo razonable, pues la clave de los métodos Monte Carlo son la repetición de las mismas y como se comenta posteriormente en 3.3, al aumentar el número de simulaciones disminuye la varianza, permitiendo mayor precisión.
- El comportamiento de las configuraciones iniciales es difícil de predecir, por lo que aquellas que crezcan sin límite podrían agotar los recursos de memoria disponibles haciendo que la ejecución sea imposible. En particular, una situación con alto consumo de memoria dificulta la ejecución de múltiples simulaciones independientes en paralelo.

Este último punto es, en nuestra opinión, el más restrictivo. Un planteamiento inicial nos podría sugerir que limitar el tamaño de la malla dos dimensional, sin embargo se perdería información en aquellas configuraciones iniciales que excedieran el tamaño fijado de la malla. Para reducir el impacto de la finitud de la malla se ha estudiado la identificación de los bordes opuestos simulando un espacio *infinito* que imita la superficie de un toro, obteniendo resultados favorables [3, 4]. Pero no es necesario lidiar con los errores derivados de este planteamiento. Una implementación en nuestra opinión más *literal* de la descripción formal del juego de vida, nos permite romper con el paradigma de la limitación de la malla. En lugar de almacenar en memoria la malla completa independientemente de su utilización, se almacenan los nodos ocupados, dados por coordenadas sobre la malla rectangular identificada con el plano cartesiano [8]. La contrapartida de esta representación es que los nodos ocupados no son los únicos sobre los que se aplican las reglas, existen nodos vacíos sobre los que también se aplican. Así la implementación del proceso de evolución se puede desglosar en dos etapas:

- Calcular todos los nodos sobre los que se van a aplicar las reglas de evolución.
- Aplicar las reglas de evolución sobre cada nodo.

El algoritmo de la primera etapa es el siguiente (Algoritmo 1). Para cada nodo ocupado se genera su vecindario y en una tabla común a todos los nodos se apunta el número de veces que aparece cada nodo del vecindario. De esta manera en la tabla resultante tenemos enumerados cada nodo que va a evolucionar y el número de nodos ocupados en su vecindario. El vecindario de un nodo ocupado con coordenadas (x, y) viene dado por el conjunto de posiciones que difieren del nodo en cuestión en una unidad como máximo:

$$\begin{array}{ccc} (x-1,\ y+1) & (x,\ y+1) & (x+1,\ y+1) \\ (x-1,\ y) & (x,\ y) & (x+1,\ y) \\ (x-1,\ y-1) & (x,\ y-1) & (x+1,\ y-1) \end{array}$$

Algorithm 1 Cálculo de los nodos sobre los que se van a aplicar las reglas de evolución

```
Require: occupiedNodes
  n ← [(−1,1), (0,1), (1,1), (−1,0), (1,0), (−1,−1), (0,−1), (1,−1)]
  for all node ∈ occupiedNodes do
    for i = 0 to 8 do
      neighborhoodNode = node + n[i]
      if neighborhoodNode ∉ table then
        table[neighborhoodNode] = 1
      else
        table[neighborhoodNode]+ = 1
      end if
    end for
  end for
  return table
```

El proceso de evolución de cada nodo almacenado en la variable *table* que devuelve el algoritmo de la etapa anterior se describe en el algoritmo 2.

Algorithm 2 Evolución síncrona

```
Require: table
Require: occupiedNodes
  nextOccupiedNodes ← []
  for all node ∈ table do
    if (table[node] = 3 or table[node] = 2) and node ∈ ocuppiedNodes then
      nextOccupiedNodes.append(node)
    else if table[node] = 3 and node ∉ ocuppiedNodes then
      nextOccupiedNodes.append(node)
    end if
  end for
  return nextOccupiedNodes
```

Notar que existen algoritmos notablemente más complejos que el descrito en esta sección, como el algoritmo *QuickLife*, el cual se utiliza en una implementación de software libre de un simulador de autómatas celulares llamado

Golly [22] o el algoritmo *HashLife*. La idea principal de este último algoritmo se basa en el reconocimiento de configuraciones más pequeñas repetidas dentro de la configuración a la que se le están aplicando las reglas de evolución [23]. Nuestra elección viene motivada por la facilidad con la que se puede implementar, describir y modificar para más tarde representar la evolución α-asíncrona.

2.2. Juego de vida de Conway α-asíncrono

Antes de nada, es importante señalar que la modificación que el juego α-asíncrono introduce en las iteraciones de actualización de los estados de los nodos de la malla, sólo afecta a éstos y no a las características de la propia malla.

En un juego de vida α-asíncrono cada nodo tiene probabilidad α de ser actualizado y probabilidad $1-\alpha$ de mantener su estado actual. Para ello se ha de generar, para cada nodo, un número pseudo-aleatorio de acuerdo a una distribución uniforme estándar. Si el número obtenido es superior a α las reglas se aplican tal y como están establecidas y, en caso contrario, no se hace, manteniendo el nodo su estado actual. Éste proceso se puede observar en detalle en el algoritmo 3. Notar que si $\alpha = 1$ el algoritmo de actualización α-asíncrono coincide con el síncrono.

Algorithm 3 Evolución α-síncrona

Require: $table$
Require: $occupiedNodes$
Require: $getRandomNumber()$
Ensure: $0 < \alpha \leq 1$
Ensure: $0 < getRandomNumber() < 1$
  $nextOccupiedNodes \leftarrow []$
  for all $node \in table$ **do**
    if $getRandomNumber() < \alpha$ **then**
      if $(table[node] = 3$ **or** $table[node] = 2)$ **and** $node \in ocuppiedNodes$ **then**
        $nextOccupiedNodes.append(node)$
      else if $table[node] = 3$ **and** $node \notin ocuppiedNodes$ **then**
        $nextOccupiedNodes.append(node)$
      end if
    end if
  end for
  return $nextOccupiedNodes$

2.3. Configuraciones iniciales del juego de vida de Conway

Como ya se ha indicado anteriormente, en este trabajo tratamos de caracterizar el comportamiento de configuraciones iniciales del juego de vida de Conway bajo la hipótesis de actualización α-asíncrona. Nuestra elección de patrones iniciales está motivada por la simplicidad de los mismos, lo que nos permite visualizar de manera sencilla el impacto del α-asincronismo.

Dado que las configuraciones iniciales son muy diversas, han existido algunos esfuerzos por realizar una taxonomía de patrones, pero no existe un consenso global. A pesar de ello, hemos considerado la existencia de tres categorías principales que, a continuación, pasamos a describir.

Vidas inmóviles

Probablemente las *vidas inmóviles* sean las configuraciones con el comportamiento más simple y fácil de observar. Esta sección está extraída de las siguientes fuentes: [24], [25] y [26].

Una *vida inmóvil* es una configuración inicial que permanece inalterada en su evolución. A continuación mostramos ejemplos de estos tipos de vidas inmóviles en la Figura 2.1. La Figura 2.1a es una *vida inmóvil* en la que todos los nodos ocupados dependen entre sí los unos de los otros, si alguno se queda libre entonces la configuración deja de serlo. Análogamente, en la Figura 2.1b la mitad horizontal derecha conserva su estabilidad gracias a su homólogo reflejado de la mitad horizontal izquierda y al alterar el estado de cualquier nodo ocupado dicha estabilidad se desvanece. Por último, la Figura 2.1c es una *vida inmóvil* y a su vez contiene a otra *vida inmóvil*.

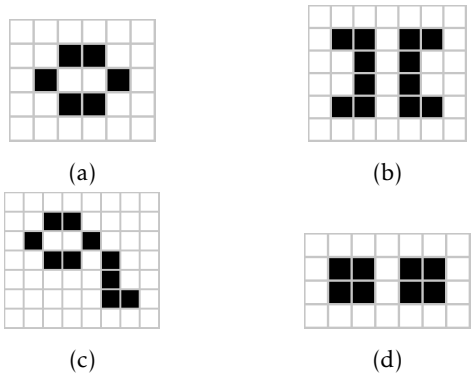


Figura 2.1: Algunas vidas inmóviles del juego de vida de Conway.

Notar que pueden estar formadas a su vez por varias *vida inmóvil* y que o

bien dependan entre sí para mantener su estabilidad como en la Figura 2.1c, o bien sean independientes y al eliminar algunas de ellas la estabilidad se mantenga como en la Figura 2.1d. También existen *vida inmóvil* que pueden ser separadas en vidas inmóviles independientes y que además existan nodos vacíos que tanto en la configuración inicial como en las *vidas inmóviles* independientes se mantengan así. Un ejemplo de esta situación se pueden observar en Figura 2.2.

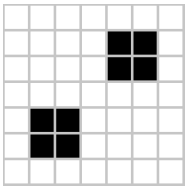


Figura 2.2: Configuración inicial que en el centro hay un nodo vacío que se mantiene tanto en las *vidas inmóviles* independientes como en el total.

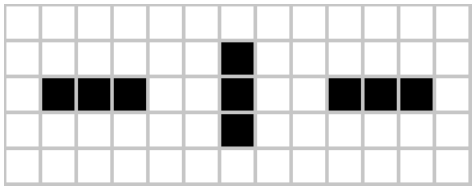
Cabría preguntarse el problema de dado un número finito de nodos ocupados, ¿cuántas *vidas inmóviles* diferentes existen? Dicho problema ha sido resuelto para *vidas inmóviles* de hasta 32 nodos ocupados, como se puede consultar en [27].

Por último, hemos observado experimentalmente que las *vidas inmóviles* se mantienen inalteradas también en el juego de vida α-asíncrono. Ya que si se actualiza alguno de sus nodos, éste permanece en el mismo estado y si no se actualiza, también. Por tanto podemos concluir que la categoría de *vidas inmóviles* no se ve alterada por la introducción de la evolución α-asíncrona. Notar que esto no quiere decir cuando una vida inmóvil colisione con otros tipos de configuraciones se mantenga inalterada. Sin embargo, existen *vidas inmóviles* que cuando colisionan con cierto tipo de naves espaciales generan la destrucción de las mismas y tras algunas iteraciones la configuración vuelve a ser inmóvil. Este tipo de vidas inmóviles se les conoce como *eaters* y existen algunos que mientras que se da la interacción con una configuración, la cual más tarde desaparece, se mantienen en todo el proceso de desaparición inalterados [28].

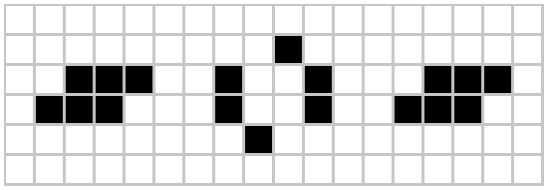
Osciladores

Un *oscilador* es una configuración inicial que tras un número fijo de iteraciones se repite en la misma posición, al número de iteraciones se le conoce como periodo del *oscilador*. En particular, las vidas inmóviles pueden ser interpretadas como *osciladores* de periodo una iteración.

En la Figura 2.3 mostramos dos configuraciones iniciales de periodo dos, durante tres iteraciones. Por un lado la Figura 2.3a muestra tres iteraciones de la configuración inicial nombrada *blinker* y por otro la Figura 2.3b son muestra iteraciones de la configuración inicial *toad*.



(a)



(b)

Figura 2.3: Primeras 3 iteraciones de 2 *osciladores* de periodo 2 del juego de vida de Conway.

Naves espaciales

Una *nave espacial* es una configuración inicial que tras un número fijo de iteraciones se repite pero en una posición desplazada. Particularmente, pueden ser vistas como osciladores que en el periodo de oscilación se desplazan. Dado que este tipo de configuraciones iniciales se desplaza sobre la malla rectangular es interesante considerar la velocidad con la que lo hacen. Si una configuración se desplaza (dx, dy) unidades cada periodo de longitud n, la velocidad de desplazamiento de la nave espacial es:

$$v = \frac{\text{máx}\{|dx|, |dy|\}}{k}$$

y su pendiente es x/y. Curiosamente se ha probado que para cada pendiente existe una *nave espacial* con dicha pendiente [29]. Notar que c es la velocidad máxima teórica, esto es, un desplazamiento por iteración.

En la Figura 2.4 encontramos dos *naves espaciales* que se desplazan en distintas direcciones. La Figura 2.5a es la *nave espacial* más pequeña conocida de velocidad $c/4$ y su desplazamiento es diagonal y la Figura 2.5b es la más pequeña conocida de velocidad $c/2$ y su desplazamiento es horizontal.

(a) (b)

Figura 2.4: Dos *naves espaciales* del juego de vida de Conway.

Almacenamiento de las configuraciones iniciales

Al igual que no existe un consenso global sobre las categorías de configuraciones del juego de vida, tampoco existe una única representación globalmente aceptada para almacenar una configuración en disco. Este problema es similar al de representar una malla rectangular infinita en una memoria finita. Existen soluciones que almacenan el rectángulo más pequeño que contiene a todos los nodos ocupados como sigue: cada fila de este rectángulo esta representada por una cadena de caracteres en la cual los nodos ocupados se identifican por un carácter y los nodos vacíos por otro. Un ejemplo de este tipo de formato es $Life1.05$ que utiliza el carácter * para los nodos ocupados y el . para los nodos vacíos.

Sin embargo, este tipo de representación puede ocupar mucho espacio en configuraciones de gran tamaño, por lo que en este trabajo hemos utilizado una versión simplificada del formato *rle* [30], el cual es más compacto que el anterior. La primera línea de este formato tiene la forma $x = m, y = n$ donde m y n son las dimensiones del rectángulo de menor tamaño que contiene a todos los nodos ocupados. La siguiente línea codifica la configuración en una secuencia de elementos de la forma número de repeticionesetiqueta, donde número de repeticiones es el número de ocurrencias de una etiqueta que puede ser alguno de los siguientes caracteres: **b** que representa un nodo vacío, **o** que representa un nodo ocupado y el carácter **$** se emplea para indicar el final de la columna del rectángulo de menor tamaño que contiene a todos los nodos ocupados. El último elemento va seguido del carácter **!** que indica el final de la configuración. El número de nodos vacíos al final de la última fila de la configuración no tiene que estar necesariamente codificado y tampoco el final de la última fila tiene que incluir al carácter **$**. La líneas iniciales de este formato que comiencen por el carácter # se interpretan como comentarios y por defecto se omiten. Por ejemplo las configuraciones de la Figura 2.1 tendrían la siguiente codificación *rle*:

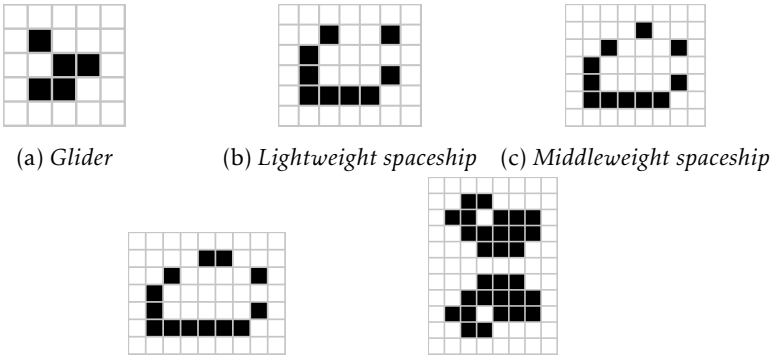
- # Configuración Figura 2.1a
  x = 4, y = 3

b2o$o2bo$b2o!

- # Configuración Figura 2.1b
  x = 5, y = 4
  2ob2o$bobo$bobo$2ob2o!
- # Configuración Figura 2.1c
  x = 6, y = 5
  b2o$o2bo$b2obo$4bo$4b2o!
- # Configuración Figura 2.1d
  x = 5, y = 2
  2ob2o$2ob2o!

Selección de configuraciones iniciales

Los primeros censos de configuraciones iniciales fueron *The Online Life-Life CA Soup Search* y *Achim Flammenkamp's census*, en los que se contabilizaron 174.631.866.050 y 50.158.095.316 configuraciones del juego de vida, respectivamente. El primero de ellos consistió en la evolución de 6.412.048.029 configuraciones iniciales aleatorias que cubren un cuadrado de lado 20 con densidad inicial de 0.5 sobre una malla rectangular infinita [31]. El segundo exploró la evolución de 1.829.196 configuraciones iniciales aleatorias sobre una malla cuadrada de lado 2048 con los bordes opuestos identificados y con una densidad inicial de 0.375 [32]. De ambos censos se puede extraer la conclusión de que las configuraciones que aparecen más a menudo son las *vidas inmóviles*, seguidas por los *osciladores* y por último las *naves espaciales*.



(a) *Glider* (b) *Lightweight spaceship* (c) *Middleweight spaceship*

(d) *Heavyweight spaceship* (e) *MWSS on MWSS*

Figura 2.5: 5 *naves espaciales* más frecuentes en el censo Catalogue.

Para nuestro trabajo tomaremos de referencia el censo más actual *Catalogue* que recoge las ejecuciones de 19.640.649.096.999 configuraciones iniciales aleatorias cuadradas de lado 16 del juego de vida de Conway. Se han obtenido un total de 429.049.899.985.558 patrones de los cuales se encontraron 161.861 tipos diferentes [33]. En la Tabla 2.1 se muestran las primeras 5 *naves espaciales* más frecuentes, de las cuales hemos seleccionado 4 para realizar nuestro experimento. Destacar que todas son de periodo 4 y que no tomaremos más dado que el número de ocurrencias disminuye drásticamente de la cuarta a la quinta posición. En la Figura 2.5 podemos observar la forma de estas configuraciones iniciales.

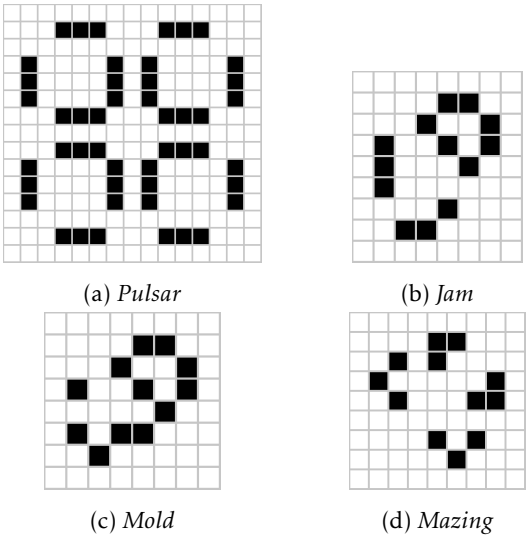
Nombre	**Periodo**	**Ocurrencias**
Glider	4	37.699.263.597.381
Lightweight spaceship	4	55.075.316.989
Middleweight spaceship (MWSS)	4	14.511.262.233
Heavyweight spaceship	4	2.521.819.486
MWSS on MWSS	4	7.077

Cuadro 2.1: *Naves espaciales* más frecuentes en el censo Catalogue.

Como representantes de la categoría de *osciladores* vamos a estudiar los dos *osciladores* más frecuentes de periodo 2, 3 y 4, cuyas frecuencias se muestran en la Tabla 2.2. Los *osciladores* más comunes de periodo 2 son *blinker* y *toad* introducidos anteriormente en la Figura 2.3. Los de periodo 3 y 4 son respectivamente *pulsar* y *jam*, y *mold* y *mazing* (Figura 2.6).

Nombre	Periodo	Ocurrencias
Blinker	2	124.127.579.342.223
Toad	2	947.040.964.637
Pulsar	3	30.512.370.641
Jam	3	676.267
Mold	4	6.731.991
Mazing	4	1.281.808

Cuadro 2.2: *Osciladores* de periodos 2, 3 y 4 más frecuentes



(a) *Pulsar* (b) *Jam*

(c) *Mold* (d) *Mazing*

Figura 2.6: *Osciladores* de periodo 3 y 4 más frecuentes en el censo Catalogue.

2.4. Simulaciones

Dado el carácter aleatorio del juego de vida α-asíncrono emplearemos los fundamentos de Monte Carlo expuestos en la sección 3.3 para medir los parámetros de interés que a continuación exponemos. Nuestras variables de interés son:

- Crecimiento de la configuración inicial: dispondremos de tres herramientas para medir el número de nodos ocupados. Estudiaremos la evolución del número de nodos ocupados en cada etapa, la evolución del área del rectángulo de menor tamaño que contenga a todas los nodos

ocupados de cada iteración y su densidad, esto es, el cociente del número de nodos ocupados por el área anterior.

- Tasa de cambio de la configuración inicial: emplearemos el concepto de calor, el número de nodos que cambian de estado por iteración.
- Distribución de los nodos ocupados en cúmulos: contabilizaremos el número de cúmulos por iteración, entendiendo por cúmulo al mayor conjunto de nodos ocupados cuyo vecindario no es disjunto, es decir, en un cúmulo cada nodo ocupado está contenido en el vecindario de otro nodo ocupado del cúmulo.

Estas variables serán medidas para distintos valores de α con el fin de estudiar el efecto de la aleatoriedad en las configuraciones iniciales. Notar que el número de simulaciones realizadas para cada patrón será variable puesto que algunos patrones son de mayor tamaño y en consecuencia tienen simulaciones más lentas.

Arquitectura de las simulaciones

Dado que realizar múltiples simulaciones de una configuración inicial toma una cantidad considerable de tiempo, hemos considerado la separación del proceso de obtención de datos de las simulaciones en tres etapas bien diferenciadas:

1. Obtención de las variables de interés para cada conjunto de simulaciones de una configuración inicial.
2. Cálculo de los promedios y sus intervalos de confianza para cada variable en cada iteración.
3. Representación gráfica de los múltiples datos obtenidos para visualizar el impacto de la α-asincronicidad en la evolución.

La primera etapa recibe como entrada:

- Número de simulaciones
- Número de iteraciones que va a realizar cada simulación
- Distintos valores de α para los que se van a realizar las simulaciones.
- Una lista de configuraciones iniciales.

Con estos datos realiza la ejecución de múltiples simulaciones con un número dijo de iteraciones de una o múltiples configuraciones iniciales. En cada iteración se calcula el número de nodos ocupados, el área del menor rectángulo que contiene a todos ellos, el calor y el número de clústeres. Adicionalmente se calcula un valor que represente unívocamente a una configuración, de manera que en caso de que en la misma iteración de múltiples simulaciones se repita un valor, éste no se almacene por duplicado. Esta etapa genera un archivo en formato *JSON* [34] con las siguientes entradas:

- *samplerSeed*: dado que cada simulación individual tiene una semilla propia, hemos decidido que dicha semilla sea generada a su vez por otro generador de números pseudo-aleatorios para asegurar una correcta distribución de las mismas. Este campo almacena la semilla que se ha empleado para el generador de semillas, asegurando la reproducibilidad de cada conjunto de simulaciones.
- *pattern*: este campo almacena el nombre de la configuración inicial sobre la que se han realizado las simulaciones.
- *alpha*: representa el valor de α-asincronismo en la evolución que se ha aplicado las simulaciones, éste valor tiene que estar entre 0 y 1 para ser un valor correcto. En particular, es el valor de α con el que se aplica el algoritmo 3 en cada iteración.
- *numberOfSteps*: este campo representa el número de iteraciones que se han realizado en cada simulación.
- *numberOfRuns*: este campo almacena el número de simulaciones que se han realizado.
- *runs*: es una lista en la cual cada posición corresponde con una iteración, a su vez cada iteración contiene un lista que representa las distintas configuraciones y las variables medidas sobre estas en la iteración. Cada elemento de la lista tiene las siguientes entradas:
  - *hash*: este valor representa unívocamente a una configuración.
  - *ncells*: este valor es el número de nodos ocupados.
  - *nclusters*: este campo es el número de clústeres calculados.
  - *area*: este valor es el área del menor rectángulo que contiene a todas los nodos ocupados.
  - *ocurrences*: este campo almacena el número de veces que se repite esta configuración, evitando almacenar duplicados.

Notar que dado que las simulaciones no comparten información entre sí, pueden ser realizadas en procesos independientes para más tarde agregar todos los resultados en una sola salida.

La siguiente etapa acepta como entrada el archivo en formato JSON de la etapa anterior y calcula el valor medio y de la desviación típica de cada variable, generando un archivo en formato CSV [35] para cada variable medida en el cual cada fila tiene tres columnas: número de iteración, media y el radio del intervalo de confianza, esto es, tres veces la desviación típica. El nombre que recibe cada archivo está compuesto por la variable medida, el valor de α, el número de simulaciones y el número de iteraciones. En la etapa final utilizaremos el nombre de los ficheros generados para interpretar su contenido adecuadamente y nombrar a su vez los gráficos de salida.

Finalmente la última etapa recibe como entrada los archivos CSV generados en la etapa anterior y realiza dos tipos de gráficos. En primer lugar realiza un gráfico para cada archivo de entrada, dónde el eje X viene dado por el número de la iteración y el eje Y por la variable observada, esta última se pinta en conjunción con su intervalo de confianza. A continuación, para cada variable medida realiza un nuevo gráfico, esta vez en lugar de representar la evolución de una variable para un solo valor de α, representa la evolución de la variable para todos los valores de α.

3 Metodología

Los formalismos nos permitirán articular la intuición de asincronismo en el esquema de actualización del autómata celular, cuyo posible homónimo biológico sería el procesamiento imperfecto de información entre individuos a causa de perturbaciones derivadas del medio o de la interacción con otros individuos. En este trabajo nos restringimos a un caso simple de asincronismo en la actualización: examinaremos que ocurre si todas las transiciones ocurren al mismo tiempo pero los individuos reciben la información del estado de sus vecinos de forma imperfecta.

En primer lugar introducimos el autómata celular m-asíncrono que nos permite articular el juego de vida de Conway α-asíncrono, a continuación exponemos los conceptos básicos de teoría de conjuntos, probabilidad y generación de números aleatorios sobre los que desarrollaremos las estimaciones Monte Carlo. Las claves de este desarrollo son el teorema central del límite y la ley de los grandes números que esencialmente justifican la efectividad del método Monte Carlo.

3.1. Teoría de la computación

Introducción

Dado que el comportamiento completamente síncrono de un autómata celular como herramienta de modelado es una rareza, se han realizado numerosas investigaciones empíricas del autómata celular asíncrono. Sin embargo, los pocos análisis formales realizados o bien se refieren a ejemplos o bien a casos particulares de asincronicidad. Tomaremos el concepto de autómata celular m-asíncrono [19], cuya idea principal es tener algún tipo de oráculo el cual en cada iteración escoge los nodos que tienen que ser actualizados. Dicho oráculo se implementa a través de una medida de probabilidad μ sobre subconjuntos de enteros d-dimensionales, $\mathbb{Z}^d$. Notar que la definición con la que trataremos es la extensión a espacios multidimensionales de la dada en [19].

Autómata celular determinista

Un autómata celular determinista es un sistema dinámico discreto consistente en un array d-dimensional de autómatas finitos, llamados nodos. Cada nodo está conectado uniformemente a un vecindario formado por un número finito de nodos, tiene un estado de un conjunto finito de estados y actualiza su estado de acuerdo a una función de transición local, la cual determina el siguiente estado de un nodo considerando su propio estado y el de su vecindario.

Definición 3.1.1. Formalmente, la tupla $A = (\mathbb{Z}^d, N, Q, f)$ es un autómata celular determinista, de ahora en adelante autómata celular, donde:

- $\mathbb{Z}^d$ es un espacio d-dimensional.
- Q el conjunto de estados posibles para cada nodo.
- $N \in (\mathbb{Z}^d)^k$ es el vecindario genérico de un autómata celular, esto es, para $N = (n_1, ..., n_k)$, $a \in \mathbb{Z}^d$ nodo, cada nodo ocupado en $\{(a + n_1, ..., a + n_k)\}$ es un nodo ocupado vecino de a.
- $f : Q^{k+1} \to Q$ es la función de transición local que define la transición de estado de cada nodo como función de su propio estado y del estado de cada nodo ocupado en su vecindario.

Definición 3.1.2. Una configuración es una función $g : \mathbb{Z}^d \to Q$, la cual a cada punto del espacio $\mathbb{Z}^d$ le asigna un estado del conjunto de estados Q, al conjunto de las configuraciones lo notaremos $Q^{\mathbb{Z}^d}$. Entenderemos por configuración inicial, a aquella a la que aún no se le ha aplicado la función de transición global.

Definición 3.1.3. La función de transición local induce una función de transición global $F : Q^{\mathbb{Z}^d} \to Q^{\mathbb{Z}^d}$ definida como sigue:

$$\forall x \in Q^{\mathbb{Z}^d}, \quad \forall i \in \mathbb{Z}^d, \quad F(x)(i) = f(x(i), x(i + n_1), ..., x(i + n_k)).$$

Nos gustaría poder plasmar la intuición de que un nodo tenga la misma probabilidad de ser actualizada independientemente o no del resto de nodos actualizados en cada iteración. El concepto de ultrafiltro nos permitirá establecer una clase de autómata celular lo suficientemente general como para abarcar los nuevos tipos de autómatas celulares que introduciremos en posteriores secciones.

Dado un conjunto X, $\mathcal{P}(X)$ denota el conjunto de todos los subconjuntos de X. Dado $A \in \mathcal{P}(X)$, notaremos su complementario A^c.

Definición 3.1.4. $U \in \mathcal{P}(X)$ es un ultrafiltro de X si:

1. $\emptyset \in U$.
2. Sean $A, B \in \mathcal{P}(X)$ tales que $A \subset B$ y $A \in U$, entonces $B \in U$.
3. Si $A, B \in U$, entonces $A \cap B \in U$.
4. Si $A \in \mathcal{P}(X)$ entonces o bien $A \in U$, o bien $A^c \in U$.

Además dado $p \in X$, el ultrafiltro U_p diremos que es principal si es el más pequeño que contiene a p, esto es, la colección de todos los conjuntos que contienen a p.

Definición 3.1.5. Un autómata celular m-asíncrono C es una tupla (A, μ) donde:

- A es un autómata celular.
- μ es una medida de probabilidad sobre la σ-álgebra de Borel en $\mathcal{P}(\mathbb{Z}^d)$.

Definición 3.1.6. Para cada función de transición local f y cada conjunto $\tau \in \mathcal{P}(\mathbb{Z}^d)$, definimos la función de transición global $F : Q^{\mathbb{Z}^d} \to Q^{\mathbb{Z}^d}$ como sigue:

$$\forall x \in Q^{\mathbb{Z}^d}, \quad \forall i \in \mathbb{Z}^d, \qquad F_\tau(x)(i) = \begin{cases} f(x(i), x(i+n_1), ..., x(i+n_k)) & si \quad i \in \tau, \\ \\ x(i) & si \quad i \notin \tau. \end{cases}$$

Es decir, F_τ aplica la función de transición local solo sobre los elementos de $\tau \subset \mathbb{Z}^d$.

Notar que cada nodo $i \in \mathbb{Z}^d$ es actualizado con probabilidad $\mu(U_i)$.

Esta nueva definición de autómata celular m-asíncrono, incluye la de autómata celular síncrono. Fijada una σ-álgebra $\mathbb{B}$ sobre $\mathbb{Z}^d$ y sea $C_0 = (A, \mu_0)$ un autómata celular m-asíncrono donde $\mu_0 : \mathbb{B} \to [0, 1]$ viene dada por:

$$\forall A \in P(\mathbb{Z}^d), \qquad \mu_0(A) = \begin{cases} 1 & si \quad \mathbb{Z}^d \in A, \\ \\ 0 & si \quad \mathbb{Z}^d \notin A. \end{cases}$$

De esta manera, $\mu_0(\{\mathbb{Z}^d\}) = 1$ y por lo tanto, en cada instante de tiempo se aplicará la función de transición local sobre $\mathbb{Z}^d$.

Por otro lado, también contiene el concepto de evolución totalmente asíncrona comentado en la introducción, esto es, en cada instante se aplica la función de transición local a un solo nodo.

Definición 3.1.7. Consideramos ahora el autómata celular m-asíncrono $C = (A, \mu_1)$ donde $\mu_1 : \mathbb{B} \to [0,1]$ verifica lo siguiente:

1. $\mu_1(U_i) > 0, \quad \forall i \in \mathbb{Z}^d$.
2. $\mu_1(U_i \cap U_j) = 0, \quad \forall i \neq j, \quad i, j \in \mathbb{Z}^d$.

Así solo los ultrafiltros de la forma $\{k\}$ $(k \in \mathbb{Z}^d)$ se les aplica la función de transición local.

Por último, contiene el concepto de evolución α-asíncrona que nos interesa.

Definición 3.1.8. Dado $C = (A, \mu_2)$ un autómata celular m-asíncrono y sea $\alpha \in (0,1)$ la probabilidad con la que se actualizan los nodos, donde $\mu_2 : \mathbb{B} \to [0,1]$ satisface:

1. $\mu_2(U_i) = \alpha, \quad \forall i \in \mathbb{Z}^d$.
2. $\forall A \subseteq \mathbb{Z}^d \quad$ finito, $\quad \mu_2(\bigcap_{a \in A} U_a) = \prod_{a \in A} \mu_2(U_a)$.

y lo notaremos $C(\alpha)$.

Juego de vida de Conway

Definición 3.1.9. El juego de vida de Conway es un autómata celular síncrono:

$$\mathbb{A} = (\mathbb{Z}^2, N, Q, f)$$

donde

- $N = \{(-1,1),(0,1),(1,1),(-1,0),(1,0),(-1,-1),(0,-1),(1,-1)\}$,
- $Q = \{0,1\}$,
- $f : \{0,1\}^9 \to \{0,1\}$ dada por:

$$f(x) = \begin{cases} 1 \quad si \quad x_0 = 0 \quad y \quad \sum_{i=1}^{8} x_i = 3 \\ 1 \quad si \quad x_0 = 1 \quad y \quad \sum_{i=1}^{8} x_i \in \{2,3\} \\ 0 \quad si \quad \sum_{i=1}^{8} x_i \notin \{2,3\} \end{cases} \tag{3.1}$$

y $x = (x_0, x_1, ..., x_8) = (c, c + n_1, ..., c + n_8)$ con $c \in \mathbb{Z}^d$ nodo.

Configuraciones iniciales

Fijada una configuración inicial $z \in Q^{\mathbb{Z}^2}$, enteremos por ejecución o simulación del juego de vida de Conway de duración $n \in \mathbb{N} \cup \{\infty\}$ al resultado de aplicar n veces la función de transición global del juego de vida de Conway a la configuración inicial y lo notaremos $\mathbb{A}_n(z)$. Sea t tal que $1 \leq t \leq n$, diremos que es el instante t de la ejecución del juego de vida de Conway de duración t, el resultado de aplicar t veces la función de transición global a una configuración inicial z y lo notaremos $\mathbb{A}_n^t(z)$. Por último, los conjuntos de puntos $\mathbb{A}_{n'}^{t'}(z')$, $\mathbb{A}_n^t(z)$ serán iguales, si los conjuntos de nodos ocupados de cada simulación lo son.

Definición 3.1.10. Una *vida inmóvil* es una configuración inicial que permanece en cada iteración, esto es, fijado $n \in \mathbb{N}$, se verifica que $z = \mathbb{A}_0^n(z) = \mathbb{A}_t^n(z)$ para todo $t \leq n$.

Definición 3.1.11. Un *oscilador* es una configuración inicial $z \in Q^{\mathbb{Z}^2}$ que se repite tras la aplicación de k veces de la función de transición global, esto es, fijado $n \in \mathbb{N}$, se verifica que existe $k > 1, \in \mathbb{N}$ tal que $\mathbb{A}_t^n(z) = \mathbb{A}_{t+k}^n(z)$ para todo $t \leq n - k$ y diremos que k es el periodo del *oscilador*.

Definición 3.1.12. Una *nave espacial* es una configuración inicial $z \in Q^{\mathbb{Z}^2}$ que se repite tras la aplicación de k veces de la función de transición global pero en una posición distinta, esto es, fijado $n \in \mathbb{N}$, se verifica que existen $k \in \mathbb{N}$ y una traslación distinta de la identidad, $\phi : \mathbb{Z}^2 \to \mathbb{Z}^2$, tal que $\mathbb{A}_t^n(z) = \phi(\mathbb{A}_{t+k}^n(z))$ para todo $t \leq n - k$ y diremos que k es el periodo de la *nave espacial*.

Juego de vida de Conway α-asíncrono

Definición 3.1.13. El juego de vida de Conway α-asíncrono es un autómata celular m-asíncrono $\mathbb{A}_\alpha = (\mathbb{A}, \mu)$ donde $\mathbb{A}$ es el autómata celular síncrono del juego de vida de Conway, $\mu : \mathbb{B} \to [0, 1]$ una medida de probabilidad verificando las condiciones expuestas en 3.1.8 y $\alpha \in (0, 1)$.

3.2. Teoría de la probabilidad

El contenido de esta sección está extraído de los siguientes textos: [36–38].

Definición 3.2.1. Una σ-álgebra, $\mathbb{F}$, sobre un conjunto X, es una colección no vacía de subconjuntos de X cerrados para uniones numerables y para la operación de complementario, esto es:

- $\forall A \in \mathbb{F}$ se verifica que $A^c \in \mathbb{F}$.
- $\forall A_n \in \mathbb{F}, n \in \mathbb{N}$ se verifica que $\bigcup_{n\in\mathbb{N}} A_n \in \mathbb{F}$.

Definición 3.2.2. Sean un conjunto X con su σ-álgebra asociada, $\mathbb{F}$, el par $(X, \mathbb{F})$ es un espacio medible.

Definición 3.2.3. Una función medible es una función entre espacios medibles, $g : (X, \mathbb{F}) \to (X', \mathbb{F}')$ tal que: $g^{-1}(A) \in \mathbb{F} \quad \forall A \in \mathbb{F}'$.

Definición 3.2.4. La tupla $(X, \mathbb{F}, P)$ es un espacio de probabilidad si:

- X es el espacio de muestreo, esto es, algún conjunto no vacío.
- $\mathbb{F}$ es una σ-álgebra de sucesos.
- $P : \mathbb{F} \to \mathbb{R}$ es una medida de probabilidad, esto es, P satisface los siguientes axiomas de Kolmogorov:
  1. Para cada $A \in \mathbb{F}$, existe un número $P(A) \geq 0$, esto es, la probabilidad del suceso A,
  2. $P(X) = 1$.
  3. Sean $A_n, n \geq 1$ disjuntos, entonces:

$$P\left(\bigcup_{n=1}^{\infty} A_n\right) = \sum_{n=1}^{\infty} P(A_n).$$

Definición 3.2.5. Los sucesos $A_n, n \geq 1$ son independientes si y solo si

$$P\left(\bigcap_{n\geq 1} A_n\right) = \prod_{n\geq 1} P(A_n).$$

Definición 3.2.6. Un conjunto A es abierto si, para cada punto $x \in A$, existe una bola de centro el punto y radio $\epsilon > 0$, $B(x, \epsilon) = \{z : |z - x| < \epsilon\}$, tal que $B(x, \epsilon) \subset F$. Así la σ-álgebra de Borel, es aquella generada por los conjuntos abiertos de $\mathbb{R}$.

De ahora en adelante supondremos que se ha fijado el espacio medible dado por $X = \mathbb{R}$ y $F = \mathbb{B}$ la σ-álgebra de Borel sobre $\mathbb{R}$.

Variables aleatorias

Definición 3.2.7. Una variable aleatoria definida sobre un espacio de probabilidad $(X, \mathbb{B}, P)$ es una función medible $A : X \to \mathbb{R}$.

Cada valor de A se corresponde con un subconjunto de puntos de X que se aplica en dicho valor: $\{w \in X : A(w) = x\}$, que notaremos por simplicidad $\{X = x\}$. A parte de los anteriores conjuntos también nos resultarán de interés lo siguientes:

$$\{w \in X : A(w) \leq x\} = \{A \leq x\}$$
$$\{w \in X : A(w) < x\} = \{A < x\}$$
$$\{w \in X : A(w) > x\} = \{A > x\}$$
$$\{w \in X : A(w) \geq x\} = \{A \geq x\}$$

Proposición 3.2.1. *Si $g : \mathbb{R} \to \mathbb{R}$ es medible y A es una variable aleatoria entonces $A' = g(A)$ es una variable aleatoria.*

Definición 3.2.8. La variables aleatorias $A_1, A_2, ..., A_n$ son independientes si y solo si, para arbitrarios conjuntos de la σ-álgebra de Borel $B_1, B_2, ..., B_n$:

$$P\left(\bigcap_{k=1}^{n}\{A_k \in B_k\}\right) = \prod_{k=1}^{n} P(A_k \in B_k).$$

Definición 3.2.9. Dada una variable aleatoria A se define su función de distribución como $F : \mathbb{R} \to [0, 1]$ dada por:

$$x \mapsto F(x) = P(A \leq x).$$

Proposición 3.2.2. *La función de distribución de la variable aleatoria A satisface:*

- *Es monótona no decreciente.*
- *Dada una sucesión decreciente de elementos de $\mathbb{R}$, $\{x_n\}_{n\in\mathbb{N}} \in \mathbb{R}$ convergente a $x \in \mathbb{R}$ se tiene $\lim_{x_n \to x} F(x_n) = F(x)$, es decir, es continua a la derecha.*
- $\lim_{x\to+\infty} F(x) = 1$ *y* $\lim_{x\to-\infty} F(x) = 0$.

Definición 3.2.10. Sea F una función de distribución definimos la función de densidad como la función integrable, f, tal que:

$$F(b) - F(a) = \int_a^b f(x)dx, \quad \forall a < b.$$

Definición 3.2.11. Sea A una variable aleatoria, definimos su valor esperado o esperanza como sigue:

$$\mathbb{E}(A) = \int_X A(w)dP(w).$$

Adicionalmente si $\mathbb{E}|A| < \infty$, diremos que A es integrable.

Definición 3.2.12. Sea una variable aleatoria A, definimos:

- Los momentos de orden n de A: $\mathbb{E}(A^n) = \int_X A(w)^n dP(w), \quad n \in \mathbb{N}$.
- Los momentos centrados de orden n de A: $\mathbb{E}_c(A^n) = E((A - E(A))^n), \quad n \in \mathbb{N}$.

Notar que los momentos no existen necesariamente para todo $n \in \mathbb{N}$.

Definición 3.2.13. Definimos la varianza de la variable aleatoria A con esperanza $\mu < \infty$ y $\mathbb{E}_c(A^2) < \infty$ como:

$$var(A) \equiv \mathbb{E}_c(A^2) = \mathbb{E}(A^2) - \mu^2.$$

Definición 3.2.14. A la raíz cuadrada positiva de la varianza la notaremos $\sigma(A) = +\sqrt{var(A)}$ y diremos que es la desviación estándar de la variable aleatoria A.

Proposición 3.2.3. *Sea A una variable aleatoria con esperanza $\mu < \infty$ y varianza $\sigma^2 < \infty$ y sea $A' = aA + b$, donde $a, b \in \mathbb{R}$, entonces:*

$$\mathbb{E}(A') = a\mu + b \quad y \quad var(A') = a^2\sigma^2.$$

Variables aleatorias discretas

Definición 3.2.15. Una variable aleatoria A diremos que es discreta si toma valores es un conjunto numerable, esto es, $\exists E = \{x_n\}_{n\in\mathbb{N}} \subset \mathbb{R}$ tal que $P(A \in E) = 1$.

Definición 3.2.16. La función de distribución de una variable aleatoria discreta A es la siguiente:

$$\forall x \in \mathbb{R}, \quad F(x) = P(A \leq x) = \sum_{x_n \in E, x_n \leq x} P(A = x_n).$$

Definición 3.2.17. Sea una variable aleatoria discreta A, definimos:

- Los momentos de orden n de A: $\mathbb{E}(A^n) = \sum_{x_m \in E} x_m^n P(A = x_m)$.
- Los momentos centrados de orden n de A: $\mathbb{E}_c(A^n) = \sum_{x_m \in E} (x_m - \mu)^n P(A = x_m)$.

El momento $n = 1$ se conoce como valor esperado o esperanza:

$$\mathbb{E}(A) \equiv \sum_{x_m \in E} x_m P(A = x_m).$$

Definición 3.2.18 (Tipos de convergencia: convergencia en probabilidad, convergencia casi segura y convergencia en distribución)**.** Sean A_n ,$n \in \mathbb{N}$ y A variables aleatorias, definimos:

- $A_n \to A$ en probabilidad, si para todo $\epsilon > 0$, $\lim_{n\to\infty} P(|A_n - A| > \epsilon) = 0$ y lo notaremos $A_n \to^P A$.
- $A_n \to A$ casi seguramente, si $P(\lim_{n\to\infty} A_n = A) = 1$ y lo notaremos $A_n \to^{c.s.} A$.
- $A_n \to A$ en distribución, si $\lim_{n\to\infty} P(A_n \leq x) = P(A \leq x), \quad \forall x \in \mathbb{R}$ donde $x \mapsto P(A \leq x)$ es una función continua y lo notaremos $A_n \to^d A$.

Definición 3.2.19. Sea i el número completo $i = \sqrt{-1}$, la extensión de la función exponencial $exp : \mathbb{R} \to \mathbb{R}^+$ al cuerpo de los números complejos es $exp : \mathbb{C} \to \mathbb{C}$ dada por:

$$z \mapsto exp(iz) = e^{iz} = \cos z + i \sin z.$$

Notar que el módulo de la exponencial compleja está acotado por la unidad :

$$|exp(iz)| = |\cos z + i \sin z| = \sqrt{\cos^2 z + \sin^2 z} = 1.$$

Esta propiedad de la exponencial compleja nos asegura la existencia para toda variable aleatoria de la siguiente definición.

Definición 3.2.20. La función característica asociada a la variable aleatoria A es la función $\phi_A : \mathbb{R} \to \mathbb{C}$, dada por:

$$\phi_A(t) = \mathbb{E}(e^{itA}).$$

Proposición 3.2.4. *Sea ϕ_A la función característica de la variable aleatoria A, entonces:*

- $\left|\phi_A(t)\right| \leq \phi_A(0) = 1.$
- *ϕ_A es una función uniformemente continua.*
- $\phi_{cA+b}(t) = e^{itb}\phi_A(ct), \quad c, b \in \mathbb{R}.$
- *Sea $A_1, A_2, ..., A_n, n \in \mathbb{N}$ una sucesión finita de variables aleatorias independientes, entonces:*

$$\phi_{\sum_{i=1}^n A_i}(t) = \prod_{i=1}^{n} \phi_{A_i}(t).$$

Además si $A_1, A_2, ..., A_n$ son idénticamente distribuidas:

$$\phi_{\sum_{i=1}^{n} A_i}(t) = \left(\phi_{A_1}(t)\right)^n.$$

- *Si $\mathbb{E}(A^k) < \infty$ su derivada k-ésima evaluada en 0 es $\phi_A^{(k)}(0) = i^k \mathbb{E}(A^k)$.*

Distribución normal

Definición 3.2.21. Diremos que una variable aleatoria A sigue una distribución normal con media $\mu < \infty$ y varianza $\sigma^2 < \infty$, $N(\mu, \sigma^2)$, si A tiene la siguiente función de densidad:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right), \quad \forall x \in \mathbb{R}.$$

Además su función característica es:

$$\phi_A(t) = e^{-\frac{t^2}{2}}.$$

Proposición 3.2.5. *Sean A, A' variables aleatorias pertenecientes a distribuciones normales con medias $\mu, \mu' < \infty$ y varianzas $\sigma_0^2, \sigma_1^2 < \infty$, $N(\mu, \sigma_0^2)$ y $N(\mu', \sigma_1^2)$ respectivamente, entonces la variable aleatoria dada por la suma $A+A'$ es una variable aleatoria que pertenece a una distribución normal $N(\mu + \mu', \sigma_0^2 + \sigma_1^2)$.*

Proposición 3.2.6. *Sea $N(\mu, \sigma^2)$ una distribución normal con media $\mu < \infty$ y varianza $\sigma^2 < \infty$, entonces el 68.27% de los valores de la distribución normal se encuentran en el intervalo $[\mu+\sigma, \mu-\sigma]$, el 95.45% en el intervalo $[\mu+2\sigma, \mu-2\sigma]$ y el 99.7% en el intervalo $[\mu+3\sigma, \mu-3\sigma]$, estos intervalos se conocen también como intervalos de confianza de la distribución normal.*

Teorema central del límite

Teorema 3.2.1. *Sean $A_1, A_2, ..., A_n$ variables aleatorias independientes, con esperanza $\mu < \infty$, varianza $\sigma^2 < \infty$ e idénticamente distribuidas. Entonces:*

$$\frac{1}{\sigma\sqrt{n}}\left(\frac{1}{n}\sum_{i=1}^{n} A_i - n\mu\right) \to^d N(0,1).$$

Previa a la demostración del teorema central del límite, introducimos las herramientas matemáticas que nos harán posible su demostración.

Teorema 3.2.2 (Teorema de Taylor)**.** *Sea $k \in \mathbb{N}$ y $f : \mathbb{R} \to \mathbb{R}$ k-veces diferenciable en el punto $a \in \mathbb{R}$. Entonces existe una función $h_k : \mathbb{R} \to \mathbb{R}$ tal que:*

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{(k)}(a)}{k!}(x-a)^k + h_k(x)(x-a)^k$$

y $\lim_{x\to a} h_k(x) = 0$.

Teorema 3.2.3 (Teorema de continuidad)**.** *Sean* $A_1, A_2, ..., A_n, n \in \mathbb{N}$ *variables aleatorias entonces:*

$$\lim_{n\to\infty} \phi_{A_n} = \phi_A(t), \quad \forall t \in \mathbb{R},$$

si y solo si

$$A_n \to^d A.$$

Demostración (Teorema central del límite). Sean $A_1, A_2, ..., A_n$ variables aleatorias independientes idénticamente distribuidas con esperanza $\mu < \infty$ y varianza $\sigma^2 < \infty$. Sea ahora

$$Z_n = \frac{A_1 + A_2 + ... + A_n - n\mu}{\sigma\sqrt{n}}.$$

Definimos una nueva variable aleatoria, Y_i, que es la versión normalizada de A_i:

$$Y_i = \frac{A_i - \mu}{\sigma}$$

Así definida, Y_i es idénticamente distribuida con esperanza y varianza:

$$E(Y_i) = 0, Var(Y_i) = 1.$$

Sea ahora $Z_n = \frac{Y_1+Y_2+...+Y_n}{\sqrt{n}}$, queremos ver que:

$$\lim_{n\to\infty} \phi_{Z_n}(t) = e^{-\frac{t^2}{2}}.$$

Procedemos a desarrollar el siguiente término:

$$\phi_{\frac{Y_1+Y_2+...+Y_n}{\sqrt{n}}}(t) = \prod_{i=1}^{n} \phi_{\frac{Y_i}{\sqrt{n}}}(t) = \left(\phi_{\frac{Y_1}{\sqrt{n}}}(t)\right)^n.$$

Aplicando el teorema de Taylor para obtener el desarrollo centrado en 0 para $k = 2$ de $\phi_{Y_1}(\frac{t}{\sqrt{n}})$:

$$\begin{aligned}\phi_{\frac{Y_1}{\sqrt{n}}}(t) &= \phi_{Y_1}(\frac{t}{\sqrt{n}}) \\ &= \phi_{Y_1}(0) + \frac{t}{\sqrt{n}}\phi'_{Y_1}(0) + \frac{t^2}{2n}\phi''_{Y_1}(0) + \frac{t^2}{n}h_2(t) \\ &= 1 + i\frac{t}{\sqrt{n}}\mathbb{E}(Y_1) - \frac{t^2}{2n}\mathbb{E}(Y_1^2) + \frac{t^2}{n}h_2(t) \\ &= 1 + 0 - \frac{t^2}{2n} + \frac{t^2}{n}h_2(\frac{t}{\sqrt{n}}),\end{aligned}$$

donde $\lim_{t\to 0} h_2(\frac{t}{\sqrt{n}}) = 0$.

Obtenemos que:

$$\left(\phi_{Y_1}(\frac{t}{\sqrt{n}})\right)^n = \left(1 - \frac{t^2}{2n} + \frac{t^2}{n}h_2(\frac{t}{\sqrt{n}})\right)^n \longrightarrow e^{-\frac{t^2}{2}}.$$

cuyo límite es la función característica de una variable aleatoria perteneciente a una distribución normal con media 0 y varianza 1, concluimos la demostración aplicando el teorema de continuidad 3.2.3. □

Ley de los grandes números

Teorema 3.2.4. *Sean A_n, $n \in \mathbb{N}$ variables aleatorias independientes, con esperanza $\mu < \infty$ e idénticamente distribuidas. Entonces el valor medio de A_n, $\bar{\mu}$, converge casi seguramente a μ:*

$$\bar{\mu} = \frac{1}{n}\sum_{n\in\mathbb{N}} A_n \to^{c.s.} \mu,$$

esto es, $P(\lim_{n\to\infty} \bar{\mu} = \mu) = 1$.

Para obtener una demostración un tanto más breve de éste teorema añadiremos la hipótesis de existencia del momento de orden 4 de A_n. Una demostración completa del teorema sin la hipótesis adicional se puede consultar en [36].

Lema 3.2.1. *Sean A_n,$n \geq 1$ variables aleatorias no negativas, entonces:*

$$E\left(\sum_{n\geq 1} A_n\right) = \sum_{n\geq 1} E(A_n).$$

Lema 3.2.2. *En las misma condiciones de la ley de los grandes números, existe una constante $K < \infty$ tal que para todo $n \geq 0$:*

$$\mathbb{E}\Big((\bar{\mu} - n\mu)^4\Big) \leq Kn^2.$$

Demostración. Sean

$$Z_k = A_k - \mu \quad y \quad T_n = Z_1 + Z_2 + ... + Z_n = \sum_{i=1}^{n} A_n - n\mu.$$

Entonces:

$$\mathbb{E}(T_n^4) = \mathbb{E}(\Big(\sum_{i=1}^{n} Z_i\Big)^4) = n\mathbb{E}(Z_1^4) + 3n(n-1)\mathbb{E}(Z_1^2 Z_2^2) \leq Kn^2.$$

donde en la segunda igualdad se ha empleado el desarrollo multinomial:

$$(x_1 + x_2 + ... + x_m)^n = \sum_{k_1+k_2+...+k_m=n} \binom{n}{k_1, k_2, ..., k_n} \prod_{1 \leq t \leq m} x_t^{k_t},$$

con

$$\binom{n}{k_1, k_2, ..., k_n} = \frac{n!}{k_1! k_2! \cdots k_n!}.$$

Dado que $\mathbb{E}(Z_k) = 0 \quad \forall k$ y la independencia de las variables Z_k, se cancelan todos los sumandos de la forma:

$$\mathbb{E}(Z_i Z_j^3) = \mathbb{E}(Z_i)\mathbb{E}(Z_j^3) = 0, \quad 1 \leq i, j \leq n, \quad i \neq j,$$
$$\mathbb{E}(Z_i Z_j Z_k Z_l) = \mathbb{E}(Z_i)\mathbb{E}(Z_j)\mathbb{E}(Z_k)\mathbb{E}(Z_l) = 0, \quad 1 \leq i, j, k, l \leq n, \quad i \neq j \neq k \neq l.$$

y siendo K adecuadamente elegida, por ejemplo $K = 4max\{\mathbb{E}(Z_1^4),\ \mathbb{E}(Z_1^2)^2\}$. □

Ya tenemos todos los rudimentos necesarios para proceder a demostrar el teorema de esta sección.

Demostración (Ley de los grandes números). Asumamos que $\mathbb{E}_c(A_n^4) < \infty \quad \forall n$, aplicando el lema anterior:

$$\mathbb{E}\Big((\bar{\mu}-\mu)^4\Big) \leq \frac{K}{n^2}.$$

Ahora sea $Y_n = (\bar{\mu}-\mu)^4, \forall n \in \mathbb{N}$ una variable aleatoria por la proposición 3.2.1 y en particular es no negativa, luego podemos aplicar el lema 3.2.1 en la siguiente cadena de igualdades:

$$\mathbb{E}\Big(\sum_{n\geq 1}(\bar{\mu}-\mu)^4\Big) = \sum_{n\geq 1}\mathbb{E}\Big((\bar{\mu}-\mu)^4\Big) \leq K\sum_{n\geq 1}\frac{1}{n^2} < \infty,$$

lo que implica:

$$\left|\sum_{n\geq 1}\Big(\bar{\mu}-\mu\Big)^4\right| = \sum_{n\geq 1}\Big(\bar{\mu}-\mu\Big)^4 < \infty \quad c.s.$$

Pero si una serie es convergente, entonces la sucesión de su término general de la serie converge a cero, por tanto:

$$\bar{\mu} \to \mu \quad c.s.$$

□

3.3. Fundamentos de las simulaciones Monte Carlo

El nombre *Monte Carlo* fue acuñado por los científicos que trabajaban en el desarrollo de armas nucleares en Los Álamos en la década de los 40 para designar una clase de métodos numéricos basados en el uso de números aleatorios. La esencia de este método reside en la invención de juegos de azar cuyo comportamiento puede ser usado para estudiar algún fenómeno de interés. Se podría pensar que el hecho de que resultados obtenidos por estos métodos estén sujetos a las leyes del azar es un problema, sin embargo, es un problema menor puesto que se puede determinar como de exactos son sus resultados y si se deseara obtener resultados más precisos, bastaría con incrementar el número de experimentos realizados. Actualmente, los métodos de Monte Carlos juegan un papel fundamental en la resolución de problemas matemáticos complejos, en los cuales, o bien los métodos de resolución analíticos o bien los métodos numéricos existentes requieren de grandes periodos de tiempo cómputo.

Curiosamente, en las definiciones de los métodos de Monte Carlo no hay referencia explícita al empleo de la capacidades de cómputo de los ordenadores, sin embargo el gran desarrollo que han experimentado éstos desde el

último tercio de siglo XX hasta nuestros días, los ha convertido en herramientas indispensables en las simulaciones Monte Carlo. La generación de números aleatorios ha experimentado también un importante crecimiento en las últimas décadas.

Generadores de números pseudo-aleatorios

Los generadores de números pseudo-aleatorios producen secuencias indistinguibles de una realmente aleatoria, es decir, no generan valores de una distribución uniforme si no que dado una semilla o valor de inicialización generan siempre la misma sucesión de números.

Un tipo generador de números aleatorios muy usado venía dado por la siguiente ecuación expresión:

$$I_{j+1} = aI_j + c \quad \text{mód } (m) \tag{3.2}$$

donde a es un entero positivo llamado multiplicador y c es un número natural llamado incremento. Para $c \neq 0$, el generador de la ecuación 3.2 es conocido por el nombre de *generador lineal congruente*. Claramente, en $n < m$ pasos la ecuación comienza a generar valores duplicados en el mismo orden. Conocido ésto, se hacían elecciones particulares de a, c y m que proporcionaran el mayor periodo posible. En [39] podemos encontrar algunos resultados notables sobre la elección parámetros a, c y m. La elección del valor inicial I_0 no es relevante, pues se generarán todos los naturales posibles entre 0 y $m-1$ antes de la primera repetición. Sin embargo no es suficiente con generar una sucesión de números de un largo periodo, además deben superar rigurosas baterías de tests empíricos que aseguren una buena distribución de las secuencias, además de la ausencia de patrones en las mismas. En el caso de los generadores lineales congruentes existe un resultado que afirma que las sucesivas n-tuplas de valores generados residen en al menos $(n!m)^{\frac{1}{m}}$ hiperplanos paralelos [40], luego esta clase de generadores no son adecuados para la generación de números aleatorios.

Dado el carácter empírico de las baterías de test, superarlas con éxito no asegura un generador de números perfecto, sino que probablemente se trate de un buen generador, ya que eventualmente con el suficiente tiempo se podría encontrar un test que no fuera superado con éxito. Por tanto nos interesaremos en tests que demuestren el mal comportamiento de un generador en un tiempo razonable.

De entre las numerosas baterías de tests existantes destacamos dos [41, 42]: *Dieharder*, que está basada en los primeros test estadísticos propuestos en *Diehard battery of tests*, incluye también los test desarrollados por el NIST

(National Institute for Standards and Technology) y la batería de tests *TestU01* [41, 42].

Nuestra elección es una variante del generador *Mersenne Twister* [43] presente en la biblioteca estándar del lenguaje de programación Python en las versiones posteriores a la 2.3 [44], y en el lenguaje de programación estadística *R* [45].

Métodos Monte Carlo

Los métodos de Monte Carlo se apoyan fundamentalmente en dos grandes resultados de la teoría de la probabilidad: la ley de los grandes números y el teorema central de límite. Ambos nos permiten describir la distribución límite de la suma de las variables aleatorias independientes, proporcionando también una estimación del error.

Definición 3.3.1. Una muestra aleatoria simple S_n, es un conjunto de $n \in \mathbb{N}$ variables aleatorias, $A_1, A_2, ..., A_n$, independientes e idénticamente distribuidas. En caso de que la media y la varianza de las variables aleatorias $A_1, A_2, ..., A_n$ sean finitas, las notaremos μ y σ^2 respectivamente.

La media de una muestra aleatoria simple S_n, $\mu_S = E(S_n) = \frac{1}{n}\sum_{i=1}^{n} A_i$ es una variable aleatoria gracias al resultado 3.2.1, si además la media μ y varianza σ de A_i son finitas, dicha variable aleatoria tiene la siguiente media y varianza:

$$E(\mu_S) = E(\frac{1}{n}\sum_{i=1}^{n} A_i) = \frac{1}{n}\sum_{i=1}^{n} E(A_i) = \mu,$$

$$var(\mu_S) = var(\frac{1}{n}\sum_{i=1}^{n} A_i) = \frac{1}{n^2}\sum_{i=1}^{n} var(A_i) = \frac{\sigma^2}{n}.$$

Luego su desviación estándar es $\sqrt{var(\mu_S)} = \frac{\sigma}{\sqrt{n}}$.

Definición 3.3.2. La variable aleatoria μ_S anteriormente definida diremos que es un *estimador* del valor esperado $E(A) = \mu$.

Definición 3.3.3. Sea μ_S un estimador de la media de una muestra aleatoria simple, S_n, de media $\mu < \infty$ y varianza $\sigma^2 < \infty$ y $\delta > 0$, la desigualdad de Chebychev es:

$$P\left(\left|\mu_S - \mu\right| \geq \sqrt{\frac{var(\mu_S)}{\delta}}\right) = P\left(\left|\mu_S - \mu\right| \geq \frac{\sigma}{\sqrt{n\delta}}\right) \leq \delta.$$

El gran resultado o *teorema fundamental de las simulaciones Monte Carlo* que podemos deducir de los anteriores, en particular es una consecuencia directa del teorema 3.2.4, el estimador G de la media de una muestra aleatoria simple, S_n, de media $\mu < \infty$ y varianza $\sigma^2 < \infty$ converge en probabilidad al valor esperado μ:

$$\forall \epsilon > 0, \quad \lim_{n\to\infty} P(|G - \mu| > \epsilon) = 0.$$

Es posible aplicar la desigualdad 3.3.3 para obtener la velocidad de convergencia respecto a n. Veamos un ejemplo de éste hecho, dado $\delta = \frac{1}{100}$:

$$P\left((G-\mu)^2 \geq \frac{100}{n}\sigma^2\right) \leq \frac{1}{100},$$

Haciendo n lo suficientemente grande, la varianza de G se hace tan pequeña como se quiera, esto es, disminuye considerablemente la probabilidad de obtener una gran desviación relativa a δ entre el valor esperado y el los valores obtenidos.

Es posible obtener un resultado más fuerte que el anterior como consecuencia del teorema 3.2.1. Existe una función de distribución de probabilidad que aproxima los valores del estimador G, esto es, cuando $n \to \infty$, el teorema central del límite afirma que asintóticamente los valores de G convergen a una distribución normal 3.2.21. Por tanto, es posible reescribir la función de distribución como sigue:

$$f(G) = \sqrt{\frac{n}{2\pi\sigma^2}} exp\left(-\frac{n(G-\mu)^2}{2\sigma}\right).$$

Cuando $n \to \infty$, el valor de G se encuentra en intervalos cada vez más estrechos centrados en $\mathbb{E}(G)$ y es posible medir la desviación en unidades de σ, es decir, el valor de G está dentro del intervalo centrado en $\mathbb{E}(G)$ de un error estándar el 68.3% de las veces, de dos errores estándar el 95.4% de las veces y de tres errores estándar el 99.7% de las veces, los intervalos son $[\mathbb{E}(G)+\sqrt{var(G)}, \mathbb{E}(G)-\sqrt{var(G)}]$, $[\mathbb{E}(G)+2\sqrt{var(G)}, \mathbb{E}(G)-2\sqrt{var(G)}]$ y $[\mathbb{E}(G)+3\sqrt{var(G)}, \mathbb{E}(G)-3\sqrt{var(G)}]$, respectivamente 3.2.6. Como comentábamos anteriormente la convergencia es asintótica por lo que inicialmente desconocemos como de grande debe de ser n para poder aplicar el teorema.

Cuando la varianza no es finita, es posible encontrar una distribución límite para G que llevará a un caso particular del teorema central del límite, en estos casos la distribución límite no será en general la distribución normal. Un estimador de la varianza de la media estimada viene dado por:

$$var(G_n) = \frac{1}{n-1}\left(\frac{1}{n}\sum_{i=1}^{n} A_i^2 - \left(\frac{1}{n}\sum_{i=1}^{n} A_i\right)^2\right)$$

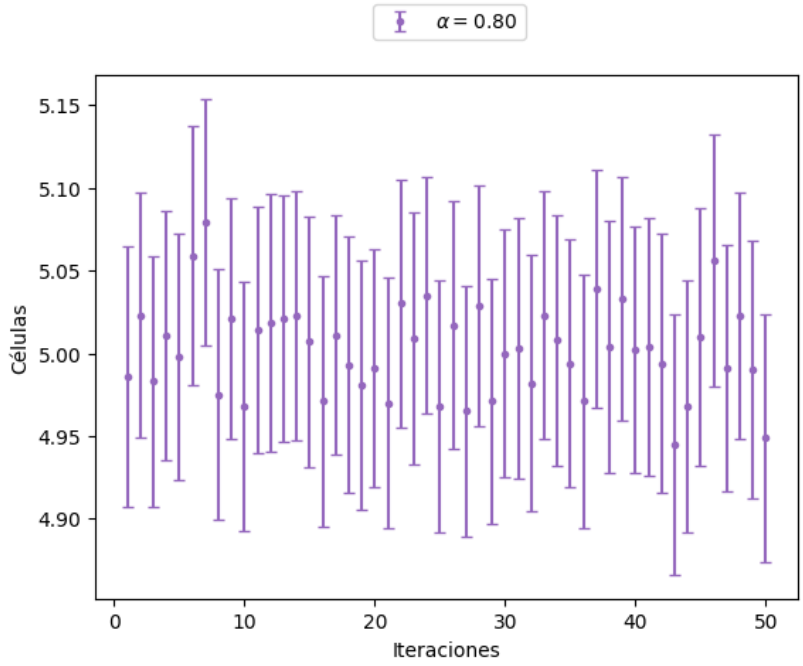
4 Análisis

Las estimaciones de los métodos Monte Carlo dependen en gran medida de la anchura del intervalo de confianza $[\mu+3\sigma, \mu-3\sigma]$. Dicha anchura se reduce con el incremento del número de muestras aleatorias pero lo hace lentamente con el consecuente incremento de tiempo de computación. Por esta razón se crean métodos alternativos conocidos como métodos de reducción de la varianza. En esta sección introducimos uno que se adecúa a nuestro problema, que utilizaremos en las todas las ejecuciones posteriores y a continuación procedemos a estudiar el efecto de la α-asincronicidad en la evolución de algunas configuraciones.

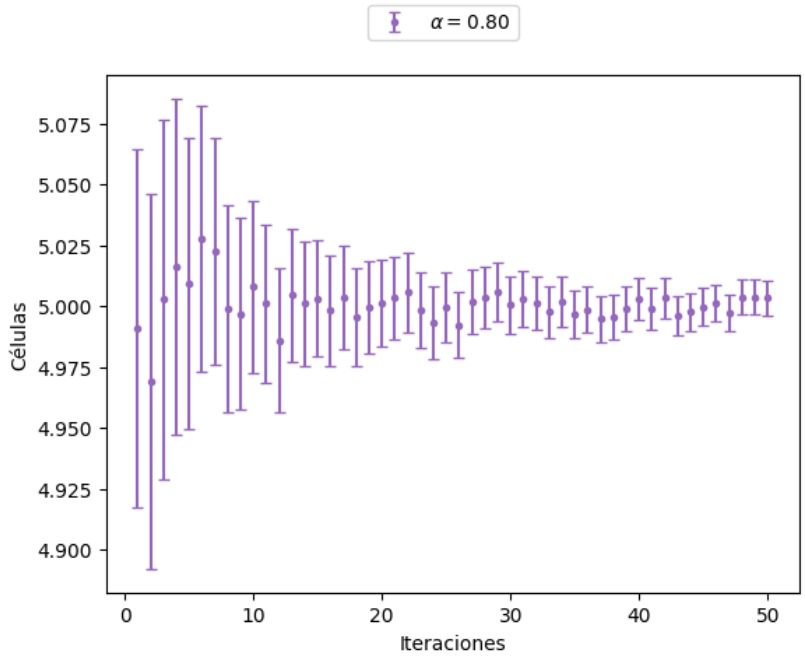
4.1. Reducción de la varianza

El intervalo de confianza $[\mu+3\sigma, \mu-3\sigma]$ crece a medida que la iteración se aleja de la configuración inicial, de decir, la varianza aumenta. Es posible reducirla aumentando el número de simulaciones globalmente e incrementando notablemente el tiempo de cálculo. Por tanto proponemos incrementar el número de simulaciones a medida que las iteraciones aumenten. Este incremento requiere de la adición de nuevas simulaciones en cada iteración, las cuales serán escogidas aleatoriamente de las ya existentes modificando la semilla para no obtener simulaciones duplicadas. Experimentalmente hemos observado que un valor que muestra buenos resultados sin aumentar excesivamente el tiempo de cálculo es un incremento en cada iteración de una décima parte del valor inicial de simulaciones.

Finalmente es posible observar la reducción del intervalo de confianza que conlleva dicho incremento en la Figura 4.1, dónde visualizamos la evolución del promedio de células en cada iteración del juego de vida. Para el contenido de esta sección no es relevante la configuración de la que se han obtenido los datos. Cada punto representa una estimación Monte Carlo junto con el intervalo de confianza $[\mu+3\sigma, \mu-3\sigma]$.



(a) Ejecución sin incremento del valor inicial de simulaciones cada iteración.



(b) Ejecución con incremento del 10% del valor inicial de simulaciones en cada iteración.

Figura 4.1: Aplicación del método de reducción de varianza.

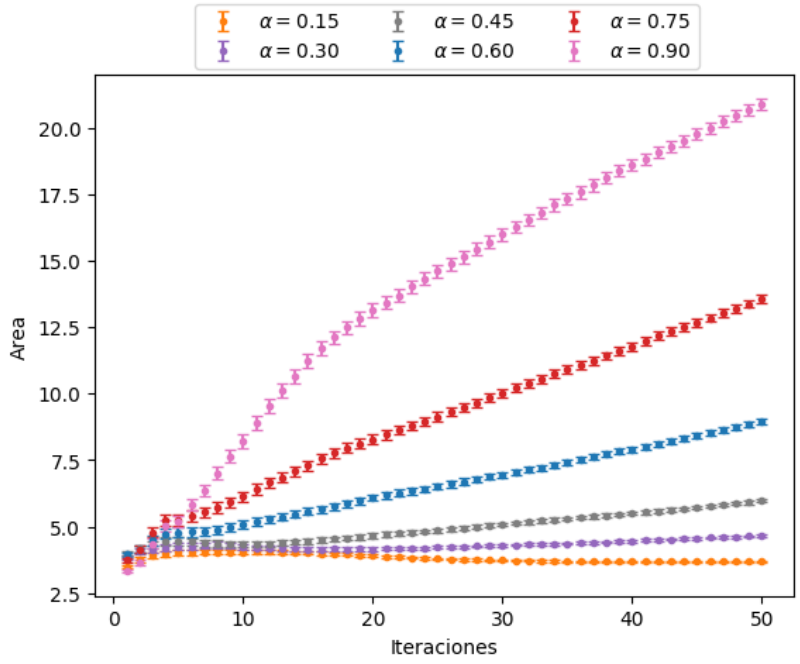
4.2. α-asincronismo en la evolución de las configuraciones iniciales

En esta sección vamos a estudiar el impacto de la α-asíncronicidad. Las configuraciones iniciales son las escogidas en la sección 2.3. Las ejecuciones son de 50 iteraciones con los valores de α: 0.15, 0.3, 0.45, 0.6, 0.75 y 0.9. Cada iteración simulada 5000 veces aplicando el método de reducción de la varianza descrito en la sección anterior. Como resultado hemos obtenido los valores medios de las variables expuestas en la sección 2.4 junto con sus correspondientes intervalos de confianza para cada iteración.

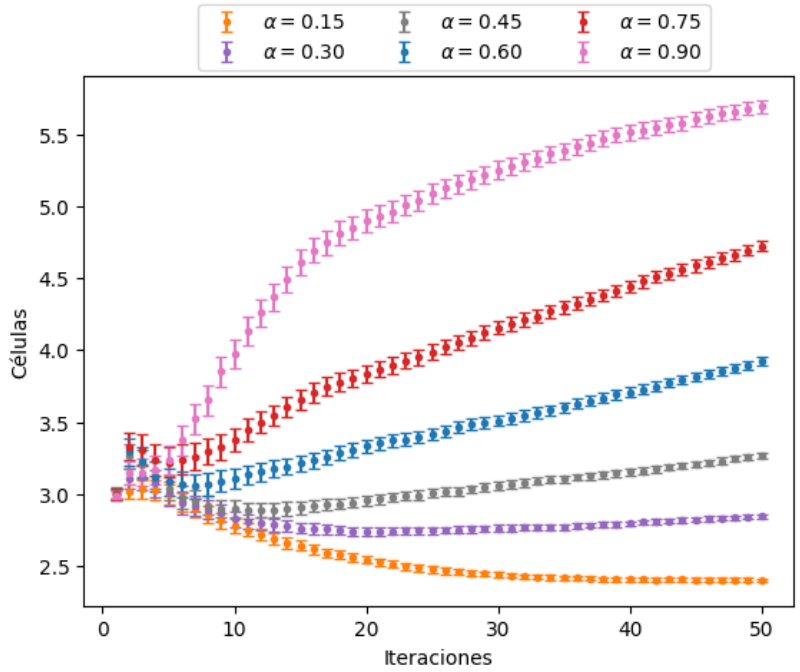
Osciladores de periodo 2

Comentamos los efectos de la variación de la α-asincronicidad sobre la configuración inicial *blinker*. Como se observa en la Figura 2.3a, cuando no existe perturbación en el intercambio de información entre células, el calor en cada iteración es 4, es decir, 2 células mueren y 2 células nacen en cada iteración, ocupa un área de 3 $nodos^2$ y está conformada por un solo clúster. En primer lugar, podemos observar en la Figura 4.2a que los valores de α se aproximan inferiormente a 0.5, imprimen un cambio de crecimiento en el área media. Mientras que para $\alpha = 0.15$, el promedio de área a partir la vigésima iteración decrece ligeramente, para $\alpha = 0.3$ el decrecimiento prácticamente desaparece, adquiriendo un valor constante y a partir de $\alpha = 0.45$ comienza una acusada tendencia de crecimiento. Esta tendencia se incrementa cuando α se aproxima a la unidad. En particular para $\alpha = 0.60$ el crecimiento a partir de la décima iteración mantiene su pendiente, hecho que no se repite para $\alpha = 0.75$, donde en la vigésima iteración cambia ligeramente la pendiente y finalmente en la décimo quinta iteración de $\alpha = 0.9$, el cambio de pendiente es más notable y desacelera marcadamente el crecimiento del área media. A la vez que estos fenómenos ocurren, la iteración a partir de la cual se produce el crecimiento del área media, decrece. Ésto es algo más visible en los valores de α superiores a 0.5.

Si observamos la Figura 4.2b, contemplamos un comportamiento similar al descrito anteriormente, por tanto resulta interesante explorar el valor medio de densidad que relaciona la variación conjunta de ambas variables.



(a)



(b)

Figura 4.2: Variación del área y número de células medio en la configuración *blinker* para distintos valores de α.

El fenómeno de decrecimiento del número de iteración a partir del cual se producen los cambios descritos anteriormente, se puede visualizar más claramente en la Figura 4.3. Cuando α se incrementa, la iteración a partir del cual la densidad se vuelve constante decrece. Por ejemplo, en la vigésima iteración $\alpha = 0.3$ el valor medio de densidad se torna constante. Otra de las cuestiones que es posible observar sobre la variación de la densidad media es que cuando α crece, el valor medio de densidad constante que se alcanza se hace cada vez menor. Ésto nos informa de que durante esas iteraciones el área crece en mayor medida que el número de células, obteniéndose valores menores de densidad.

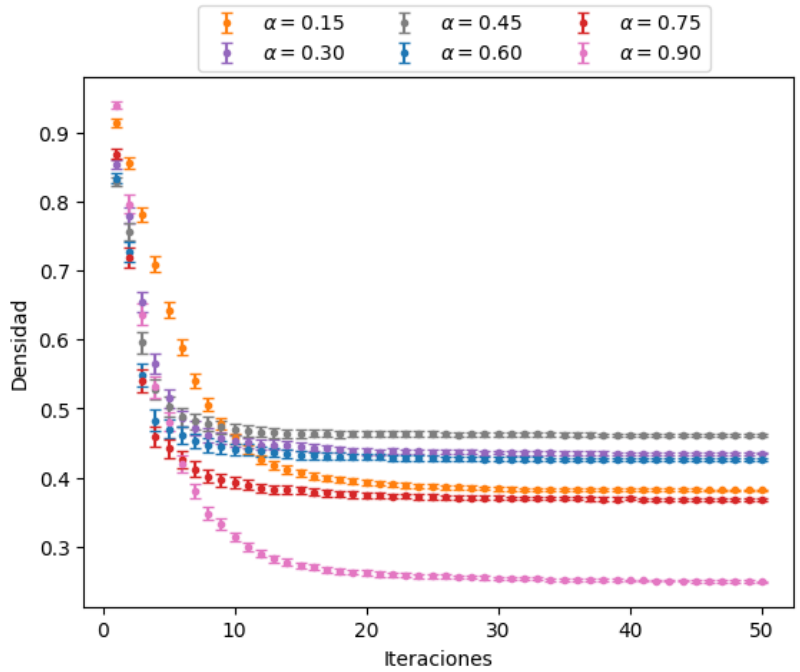
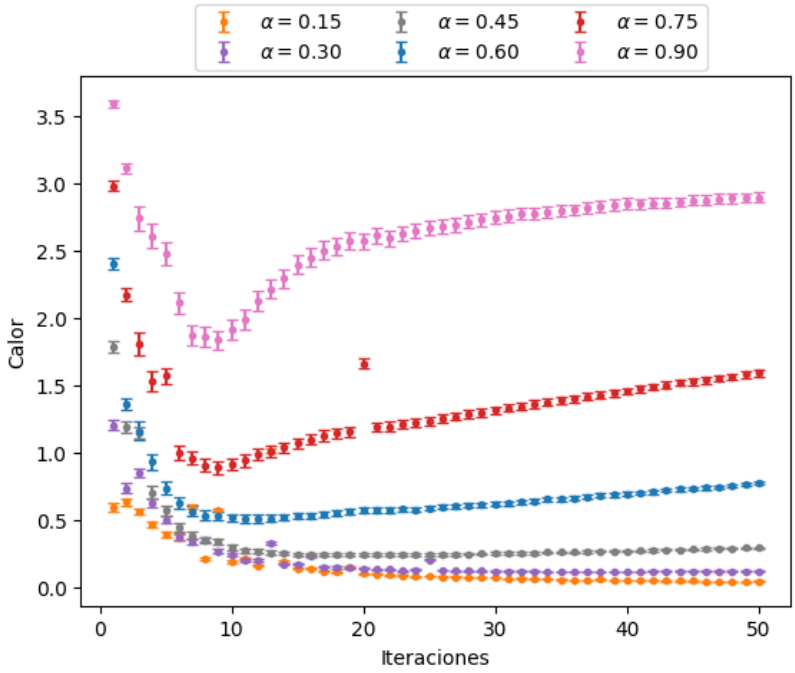


Figura 4.3: Evolución de la densidad media de la configuración *blinker* para distintos valores de α.

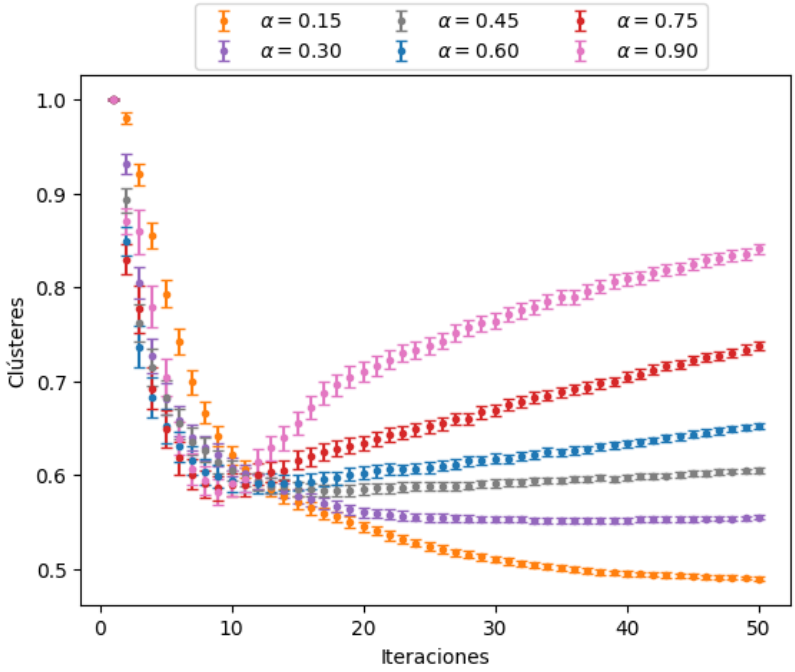
La Figura 4.4a muestra la variación del calor medio para distintos valores de α. Se pueden observar dos zonas de cambio respecto a α, una dada por los valores de α menores a 0.5 y otra por los valores superiores. La primera se caracteriza por un fuerte descenso del calor hasta el rango de iteraciones 15-20, a continuación o bien se estabiliza ($\alpha = 0.45$) o bien decrece ligeramente ($\alpha = 0.15, 0.30$). La segunda zona decrece en mucho menor medida que la anterior, hasta el rango de iteraciones 10-15 y después para $\alpha = 0.6$ crece ligera-

mente con una pendiente constante. A diferencia de los valores $\alpha = 0.75, 0.90$ que la tienen más pronunciada de la décima a la vigésima iteración y a partir de esta última, cambia frenando el desacelerando del valor medio. De hecho, el calor promedio para $\alpha = 0.9$ sugiere la existencia de una asíntota vertical con un calor medio igual a 3 nodos. A diferencia de los valores medios anteriormente comentados, el crecimiento que experimenta el calor cuando es mucho menos marcado y no llega a superar los valores iniciales.

Si consultamos la Figura 4.4b que recoge la variación del número de clústeres medio respecto de α, observamos la diferencia principal con la Figura 4.4a es que para valores de α mayores o iguales a 0.45 el promedio de clústeres decrece en aproximadamente la décima iteración a un valor cercano a 0.6, a partir de esta iteración el comportamiento es muy similar al del calor medio. En particular, el cambio de pendiente a partir de la décima iteración se reconoce unicamente para $\alpha = 0.9$, mientras que para $\alpha = 0.45, 0.60, 0.75$ la pendiente es prácticamente constante.



(a)



(b)

Figura 4.4: Variación del calor y número de clústeres medio en la configuración *blinker* para distintos valores de α.

El otro oscilador de periodo dos que hemos estudiado, la configuración *toad* (Figura 2.3b) formada por 6 células que ocupan un área de 8 $nodos^2$ en las iteraciones pares y 16 $nodos^2$ en las impares, muestra un comportamiento muy similar a *blinker* frente a la introducción de α-asíncronismo en su evolución. Por ejemplo, en la Figura 4.5 se puede apreciar como el comportamiento es profundamente similar al mostrado en la Figura 4.2b.

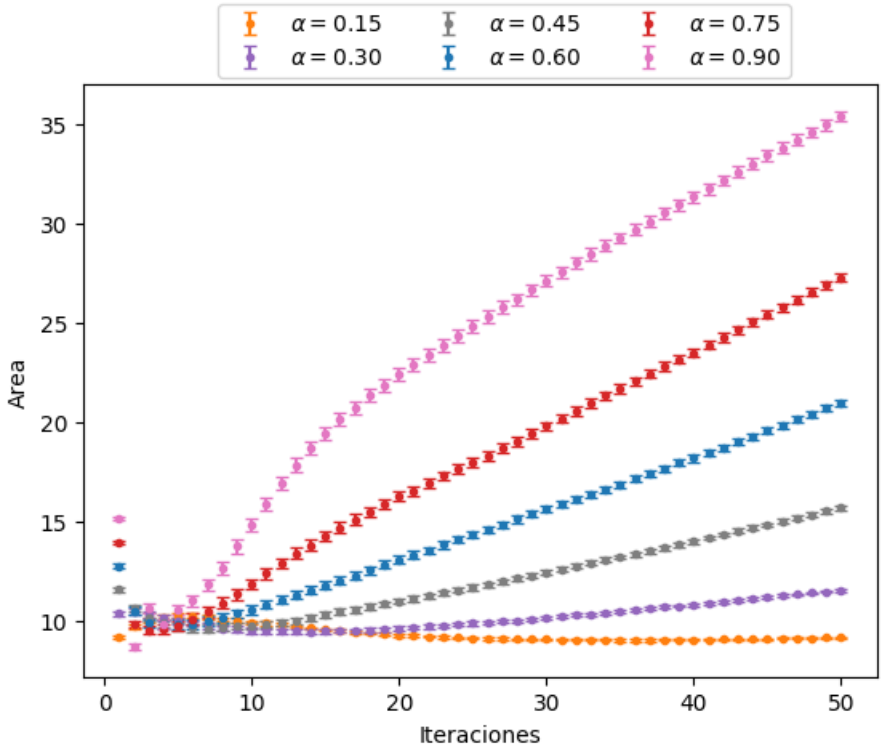


Figura 4.5: Variación del área media en la configuración *toad* para distintos valores de α.

Osciladores de periodo 3

Osciladores de periodo 4

Naves espaciales

Hasta ahora solo hemos descrito el comportamiento de *osciladores*. Como se expuso en la sección 2.3, las *naves espaciales* pueden ser vistas como *osciladores* que se desplazan, luego es interesante explorar si los comportamientos

que hemos observado en las configuraciones iniciales anteriores se reproducen en este tipo de configuraciones iniciales.

Comenzamos la sección estudiando la configuración inicial *lightweight spaceship* (Figura 2.5b). En la Figura 4.14 se muestran 4 iteraciones de esta configuración inicial. Se trata de una configuración de velocidad $c/2$ que se desplaza paralelamente al eje horizontal con un área constante de 20 $nodos^2$. En las iteraciones pares tiene 9 nodos ocupados agrupados en dos clústeres con un calor de [MISSING VALUE!] nodos y en las iteraciones impares tiene 12 nodos agrupados en un solo clúster con un calor de 8 nodos.

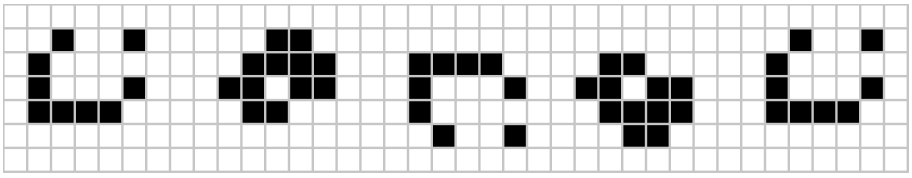
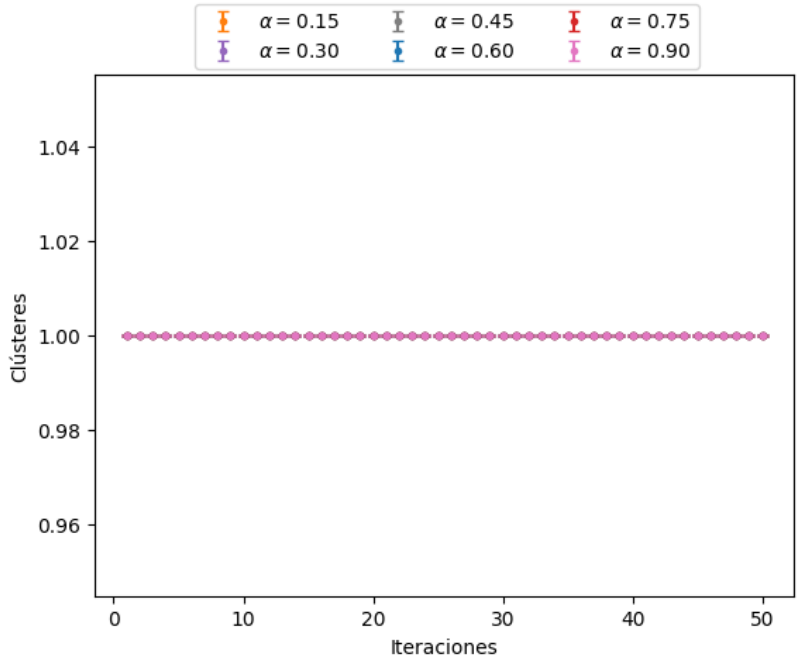


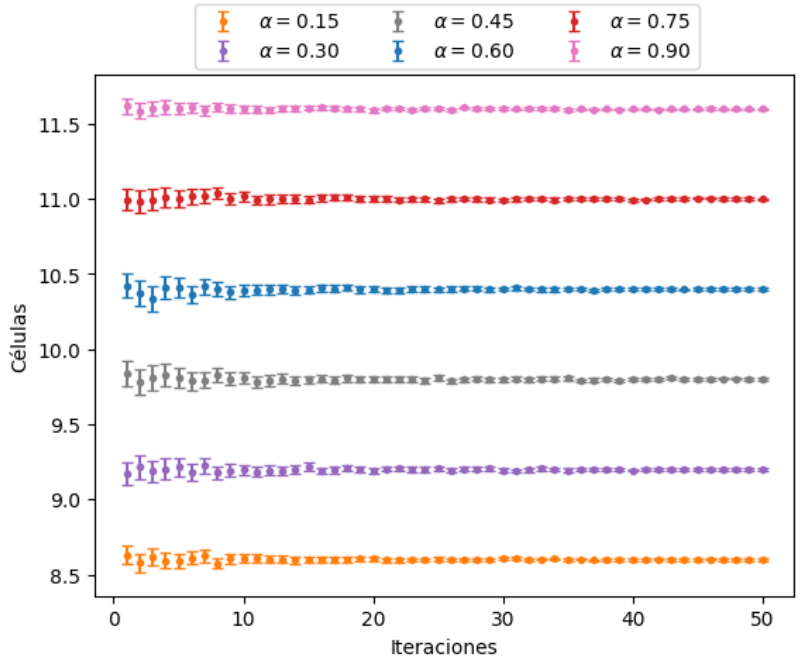
Figura 4.6: De derecha a izquierda, evolución síncrona de la configuración *lightweightspaceship*.

Uno de los característicos efectos que produce la introducción de α-asincronismo en esta configuración inicial es que el número de clústeres permanece constante durante en todas la ejecución (Figura 4.7a). Ésto contrasta con el hecho de que el resto de variables observadas tomen un valor diferente para cada valor de α. Si observamos el número medio de nodos ocupados(Figura 4.7b) se observa que tras superar la décimo quinta iteración los valores medios se estabilizan y la separación vertical de dichos valores constantes es aproximadamente igual a 0.5 nodos ocupados entre valores consecutivos de α. De esta manera el observamos que cuando el valor de α el promedio nodos ocupados se acerca a los 12 nodos ocupados en la iteraciones impares de esta configuración con la evolución síncrona.

Mientras que la densidad y el calor medios de esta configuración tienen un comportamiento similar al número medio de nodos ocupados se puede observar una variación diferente en el área media (Figura 4.8). Cuando α se aproxima, tanto inferior como superiormente, a 0.5 los valores medios de área se estabilizan entorno al valor 21.2 $nodos^2$. Por otro lado cuando α se aproxima a 0 ó a 1, los valores medios se aproximan a 20.5 $nodos^2$, un valor muy cercano a los 20 $nodos^2$ de la configuración con evolución síncrona. Otra cuestión notable es que el valor de área media pertenece en el intervalo $(20.5, 21.5)$ cuando α varía, un intervalo de pequeña longitud. A diferencia de la longitud de los intervalos en los que se encuentran los promedios de nodos ocupados, densidad y calor, que es mayor.



(a)



(b)

Figura 4.7: Variación del número medio de clústeres y de nodos ocupados en la configuración *lightweight spaceship* para distintos valores de α.

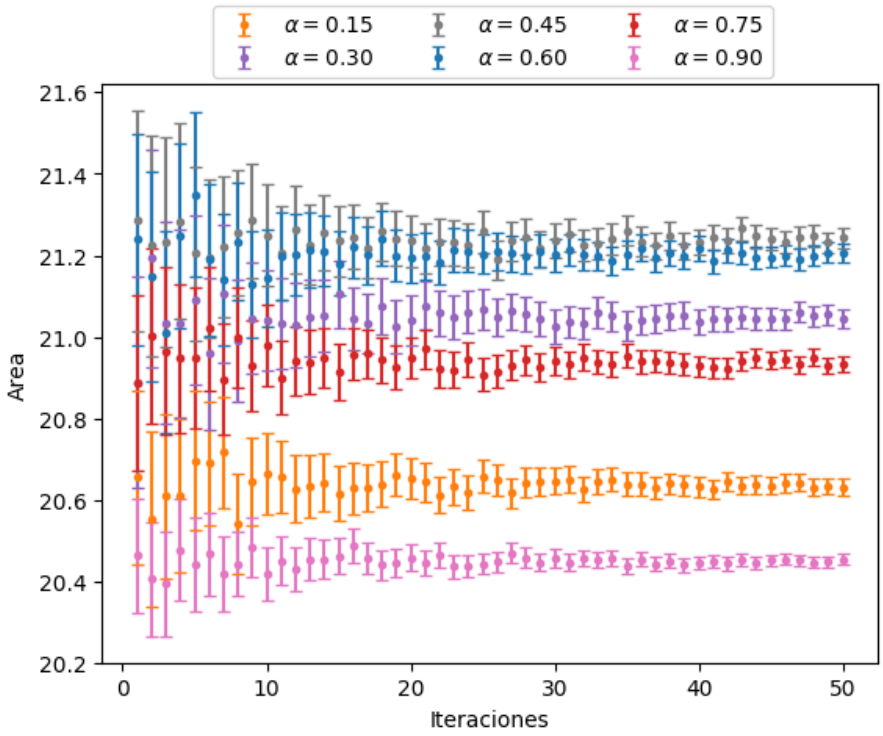


Figura 4.8: Variación del área media en la configuración *lightweight spaceship* para distintos valores de α.

La configuración inicial *middleweight spaceship* (Figura 2.5c) tiene un aspecto y una evolución síncrona similar a la configuración *lightweight spaceship*. En la Figura 4.9 se muestran 4 iteraciones de esta configuración inicial. Se trata de una configuración de velocidad $c/2$ que se desplaza paralelamente al eje horizontal con un área constante de 24 $nodos^2$. En las iteraciones pares tiene 11 nodos ocupados agrupados en tres clústeres con un calor de [MISSING VALUE!] nodos y en las iteraciones impares tiene 15 nodos agrupados en un solo clúster con un calor de [MISSING VALUE!] nodos.

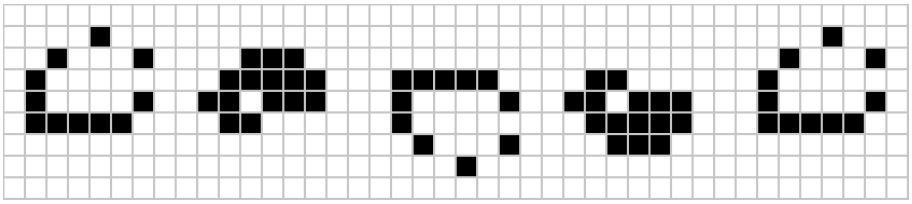


Figura 4.9: De derecha a izquierda, evolución síncrona de la configuración *middleweight spaceship*.

Al igual que en la configuración anterior, *middleweight spaceship*, el número de clústeres permanece constante independientemente del valor de α. Y tanto el promedio de nodos ocupados como el de densidad tienen el mismo comportamiento. Sin embargo para el valor medio de área se produce una variación del comportamiento que se puede observar en la Figura 4.10. El área varía prácticamente de la misma manera para $\alpha = 0.15, 0.75$ y a excepción de $\alpha = 0.90$ el resto de valores de α muestran un comportamiento constante muy parecido entre sí a partir de la vigésima iteración. Algo que sí coincide con el comportamiento de la configuración anterior es que $\alpha = 0.90$ es el promedio de área que más cercano de sitúa del valor obtenido en las iteraciones de la evolución síncrona. Otra diferencia notable es que la longitud del intervalo en el que varía el promedio crece aproximadamente una unidad.

La siguiente configuración que vamos a estudiar, *heavyweight spaceship* muestra un aspecto similar a las dos anteriores tanto en forma como en su evolución síncrona. En la Figura 4.11 se muestran 4 iteraciones de esta configuración inicial. Se trata de una configuración de velocidad $c/2$ que se desplaza paralelamente al eje horizontal con un área constante de 35 $nodos^2$. En las iteraciones pares tiene 13 nodos ocupados agrupados en tres clústeres con un calor de [MISSING VALUE!] nodos y en las iteraciones impares tiene 18 nodos agrupados en un solo clúster con un calor de [MISSING VALUE!] nodos.

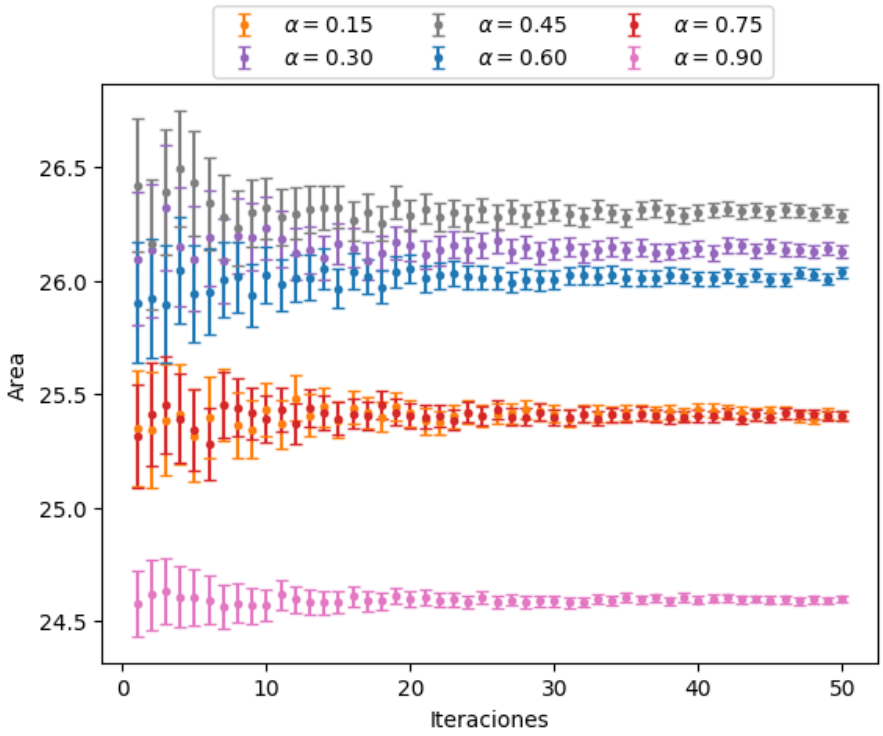


Figura 4.10: Variación del área media en la configuración *middleweight spaceship* para distintos valores de α.

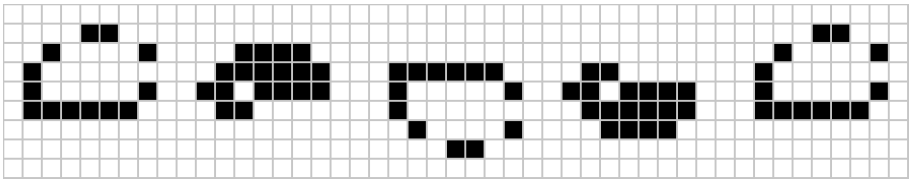


Figura 4.11: De derecha a izquierda, evolución síncrona de la configuración *heavyweight spaceship*.

A diferencia de las anteriores *naves espaciales*, en la configuración *heavyweight spaceship* el número de clústeres no se mantiene constante independientemente del valor que α tome. En la Figura 4.12 es posible observar este cambio, además todos los valores medios se mantienen aproximadamente constantes a partir de la décima iteración. A medida que α incrementa se visualiza que la separación vertical de los valores constantes del promedio de

clústeres no es constante como en las *naves espaciales* anteriores (Figura 4.7b) y dicha separación incrementa con el valor de α.

Por otra parte, la densidad media también experimenta un comportamiento diferente en situación de α-asincronismo en la evolución (Figura 4.13). De nuevo a partir de la décima iteración los promedios se mantienen aproximadamente constantes pero ahora en lugar de decrecer con el crecimiento de α, crecen de algo menos de 0.35 a algo más de 0.55 $nodos^{-}1$.

El desarrollo en situación de α-asincronismo que experimenta el valor medio de área es un comportamiento mucho más acentuado que los anteriores. Esto es, el intervalo en el que oscila el área media para distintos valores de α crece hasta una longitud de 8 $nodos^{-}1$, siendo para $\alpha = 0.9$ el menor valor de área, 31 $nodos^2$ y para $\alpha = 0.30, 0.45$ los mayores valores medios, aproximadamente 39 $nodos^2$.

Los promedios de calor y número de nodos ocupados muestran un comportamiento idéntico al que se puede observar en la Figura 4.7b.

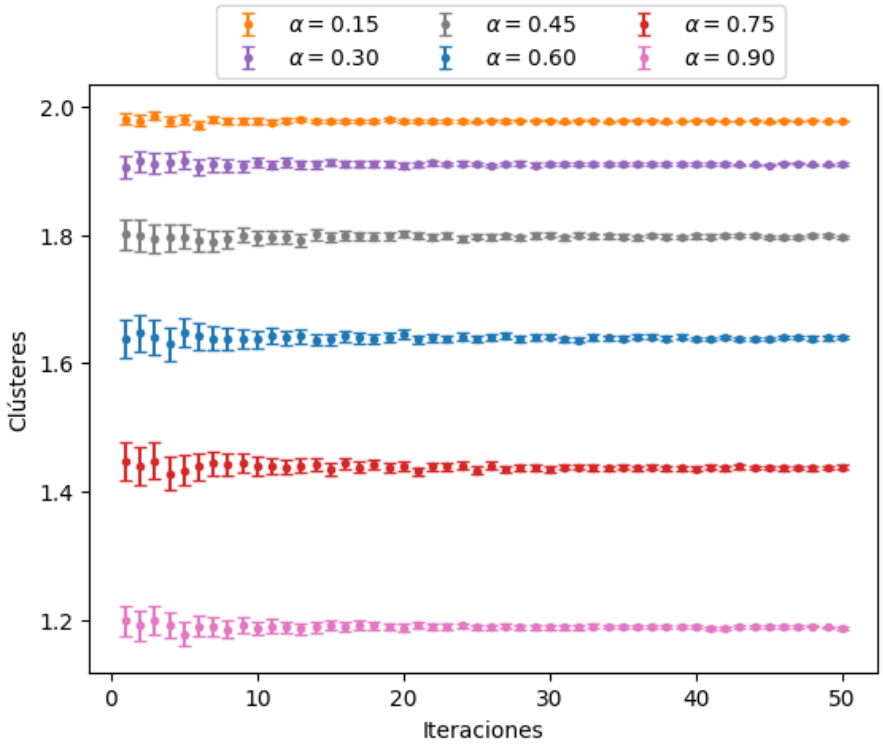


Figura 4.12: Variación del promedio de clústeres en la configuración *heavyweight spaceship* para distintos valores de α.

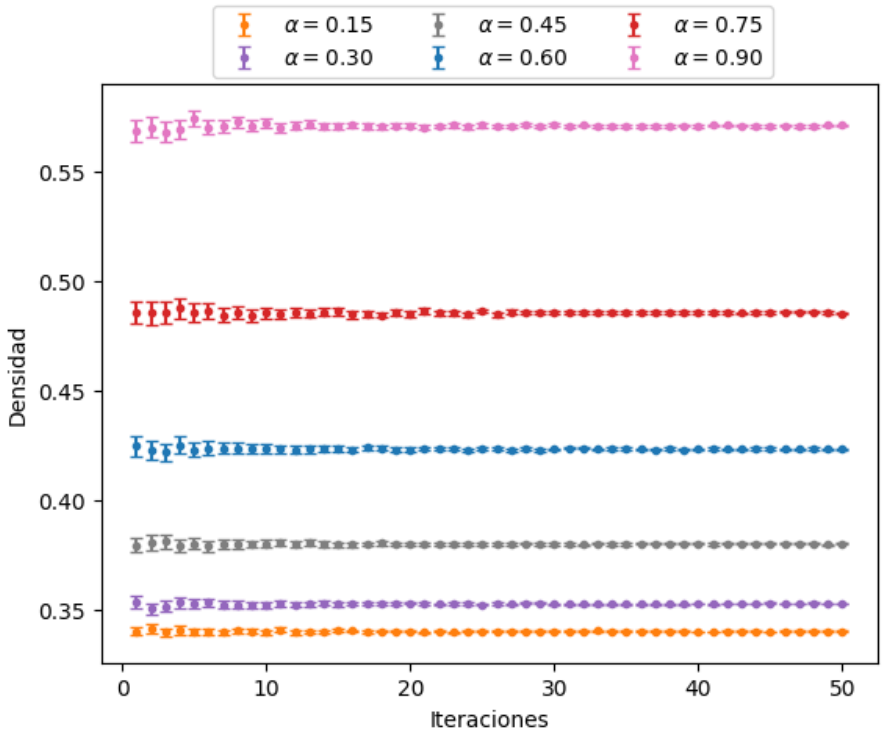


Figura 4.13: Variación del promedio de densidad en la configuración *heavy-weight spaceship* para distintos valores de α.

Por último, la configuración inicial *glider* es una nave espacial de velocidad $c/4$ formada por 5 nodos ocupados agrupados en un único clúster, tiene un calor de 4 *nodos* y ocupa un área de 9 $nodos^2$ (Figura 2.4a). Esta configuración inicial desarrolla un comportamiento sorprendentemente similar al descrito para los osciladores de periodo 2 en la sección 4.2.

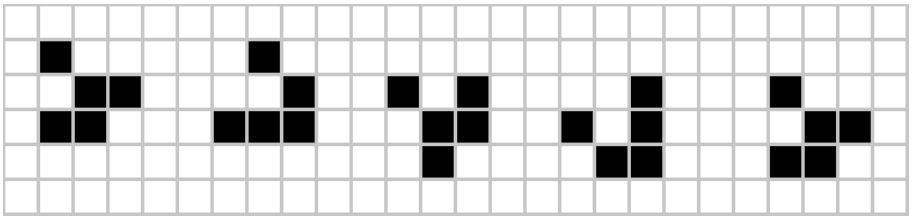


Figura 4.14: Evolución síncrona de la configuración *glider*.

5 Conclusiones

5.1. Teoría de conjuntos

Introducción

Bibliografía

[1] Edward F Moore. Machine models of self-reproduction. 14(1962):17–33, 1962.

[2] Martin Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, 223:120–123, 1970.

[3] Jan Hemmingsson. Consistent results on 'life'. *Physica D: Nonlinear Phenomena*, 80(1-2):151–153, 1995.

[4] Hendrik J Blok and Birger Bergersen. Effect of boundary conditions on scaling in the "game of life". *Physical Review E*, 55(5):6249–6252, 1997.

[5] Carter Bays. Cellular automata in the triangular tessellation. *Complex Systems*, 8(2):127, 1994.

[6] Nick Owens and Susan Stepney. Investigations of game of life cellular automata rules on penrose tilings: Lifetime, ash, and oscillator statistics. *J. Cellular Automata*, 5(3):207–225, 2010.

[7] Clifford A Reiter. The game of life on a hyperbolic domain. *Computers & Graphics*, 21(5):673–683, 1997.

[8] Conway's Game of Life - Boardless approach. https://web.archive.org/web/20190212184430/https://rosettacode.org/wiki/Conway's_Game_of_Life#Boardless_approach. [Online; accessed 14-February-2019].

[9] Stephen Wolfram. *Celular Automata and Complexity*. Addison-Wesley, 1994.

[10] Harold V McIntosh. Wolfram's class iv automata and a good life. *Physica D: Nonlinear Phenomena*, 45(1-3):105–121, 1990.

[11] Paul Rendell. *Turing universality of the game of life*. Springer, 2002.

[12] A Turing Machine in Conway's Game of Life, extendable to a Universal Turing Machine. https://web.archive.org/web/20190118023640/

http://rendell-attic.org/gol/tm.htm. [Online; accessed 14-February-2019].

[13] Build a working game of Tetris in Conway's Game of Life - StackExchange. https://web.archive.org/web/20190120082452/https://codegolf.stackexchange.com/questions/11880/build-a-working-game-of-tetris-in-conways-game-of-life. [Online; accessed 14-February-2019].

[14] Jean-Philippe Rennard. *Implementation of logical functions in the Game of Life*. Springer, 2002.

[15] Nazim A Fatès and Michel Morvan. An experimental study of robustness to asynchronism for elementary cellular automata. *arXiv preprint nlin/0402016*, 2004.

[16] Birgitt Schönfisch and André de Roos. Synchronous and asynchronous updating in cellular automata. *BioSystems*, 51(3):123–143, 1999.

[17] Birger Blok, Hendrik J.; Bergersen. Synchronous versus asynchronous updating in the "game of life". *Physical Review E*, 59:3876–3879, 1999.

[18] Nazim Fates. Critical phenomena in cellular automata: perturbing the update, transitions, the topology. *Acta Physica Polonica B*, 3:315–325, 2010.

[19] Alberto Dennunzio, Enrico Formenti, Luca Manzoni, and Giancarlo Mauri. m-asynchronous cellular automata: from fairness to quasi-fairness. *Natural Computing*, 12(4):561–572, 2013.

[20] Martin Gardner. Wheels, life, and other mathematical amusements. 86, 1983.

[21] David Eppstein. *Growth and decay in life-like cellular automata*. Springer, 2010.

[22] Quicklife algorithm implementation. https://sourceforge.net/p/golly/code/ci/master/tree/gollybase/qlifealgo.cpp. [Online; accessed 2-Jun-2019].

[23] R Wm Gosper. Exploiting regularities in large cellular spaces. *Physica D: Nonlinear Phenomena*, 10(1-2):75–80, 1984.

[24] Noam David Elkies. The still-life density problem and its generalizations. *Voronoi's Impact on Modern Science, Book I*, 1998.

[25] Matthew Cook. Still life theory. *New Constructions in Cellular Automata*, 226, 2003.

[26] Still Life - lifewiki. http://www.conwaylife.com/wiki/Still_life. [Online; accessed 21-May-2019].

[27] Number of stable n-celled patterns still lifes in Conway's game of Life. https://oeis.org/A019473. [Online; accessed 21-May-2019].

[28] Game of life lexicon. http://www.conwaylife.com/ref/lexicon/lex_e.htm#eater. [Online; accessed 2-Jun-2019].

[29] ER Berlekamp, JH Conway, and RK Guy. What is life?, chapter 25. *Winning Ways for Your Mathematical Plays*, 2, 1982.

[30] Run length encoded - LifeWiki. http://www.conwaylife.com/wiki/Run_Length_Encoded. [Online; accessed 2-Jun-2019].

[31] The Online Life-Like CA Soup Search - lifewiki. http://www.conwaylife.com/wiki/Nathaniel%27s_census. [Online; accessed 21-May-2019].

[32] Achim Flammenkamp's census - LifeWiki. http://www.conwaylife.com/wiki/Achim_Flammenkamp%27s_census. [Online; accessed 21-May-2019].

[33] Catagolue database. https://catagolue.appspot.com/home. [Online; accessed 2-Jun-2019].

[34] The JSON Data Interchange Syntax. http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf. [Online; accessed 2-Jun-2019].

[35] Common Format and MIME Type for Comma-Separated Values (CSV) Files. https://tools.ietf.org/html/rfc4180. [Online; accessed 2-Jun-2019].

[36] Allan Gut. *Probability: a graduate course*, volume 75. Springer Science & Business Media, 2013.

[37] David Williams. *Probability with martingales*. Cambridge Mathematical Textbooks. Cambridge University Press, 1991.

[38] Michel Loeve. *Probability theory I*. Springer, 4th edition, 1977.

[39] Donald E. Knuth. *The art of computer programming II*. Addison-Wesley Professional, 3 edition, 1997.

[40] George Marsaglia. Random numbers fall mainly in the planes. *Proceedings of the National Academy of Sciences of the United States of America*, 61(1):25, 1968.

[41] Dieharder: A Random Number Test Suite. http://web.archive.org/web/20190402181134/http://webhome.phy.duke.edu/~rgb/General/rand_rate.php. [Online; accessed 02-April-2019].

[42] Pierre L'Ecuyer and Richard Simard. Testu01: Ac library for empirical testing of random number generators. *ACM Transactions on Mathematical Software (TOMS)*, 33(4):22, 2007.

[43] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

[44] Python 2.7.16 Documentation. http://web.archive.org/web/20170907114211/https://docs.python.org/2.7/library/random.html. [Online; accessed 29-April-2019].

[45] R Documentation, random number generation. https://web.archive.org/web/20190106064055/https://stat.ethz.ch/R-manual/R-devel/library/base/html/Random.html. [Online; accessed 29-April-2019].