SIMULACIÓN DE SISTEMAS
Práctica 2:
Modelos de Monte Carlo. Generadores de datos

UNIVERSIDAD DE GRANADA

Sergio Aguilera Ramírez

26 de octubre de 2019

Índice

Capitulo 1: Mi Segundo Modelo de Simulación de Monte Carlo

1.1) - Modelización por Monte Carlo

En esta sección vamos a realizar pruebas sobre las diferentes distribuciones aportadas en el guión de prácticas. En las tres secciones se ha utilizado el rango [0,100] para la asignación del valor de s. Además, los valores utilizados para "veces"son 100, 1000, 5000 y 100000. En las secciones se mostrarán gráficas que representarán las salidas obtenidos por el programa mediante los diferentes valores de los parámetros utilizados.

a) - Distribución A (U(0,99))

En este experimente vamos a distribuir de forma uniforme la probabilidad de demanda. Las siguientes gráficas muestran la evolución de la ganancia a lo largo de la ejecución donde el valor de 's' se va modificando.

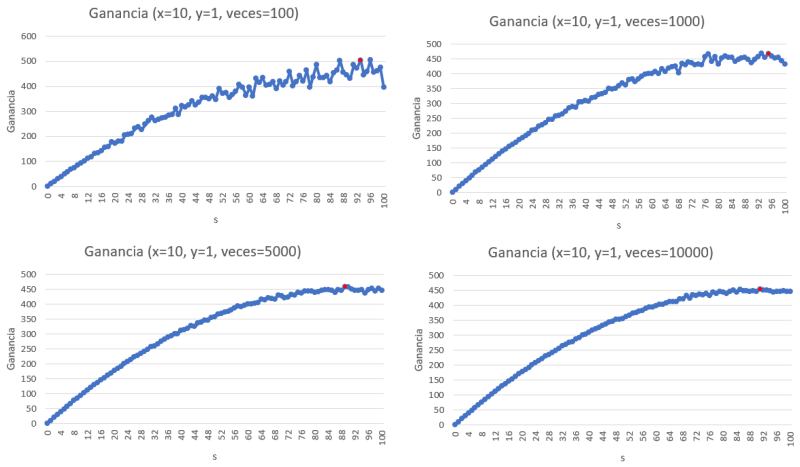

Figura 1: Evolución de la Ganancia (x=10, y=5) Distribución A

La Figura 1 muestra la evolución de la ganancia para un valor de x=10 y un valor de y=1, es decir, el valor de ganancia de cada producto es de 10 y la perdida por producto no vendido es de 1. vemos como las gráficas son crecientes hasta llegar a un cierto valor de 's' donde las gráficas se van estabilizando.

La siguiente gráfica (Figura 2), como se ha dicho en el apartado anterior muestra la evolución de la ganancia pero con un valor de x=10 y de y=5, lo cual supone una mayor perdida sobre los productos no vendidos. Como vemos cuando 's' crece mucho la ganancia empieza a caer, esto se debe a que los productos a partir de una cierta cantidad se dejan de vender lo cual supone un gran impacto sobre la ganancia.

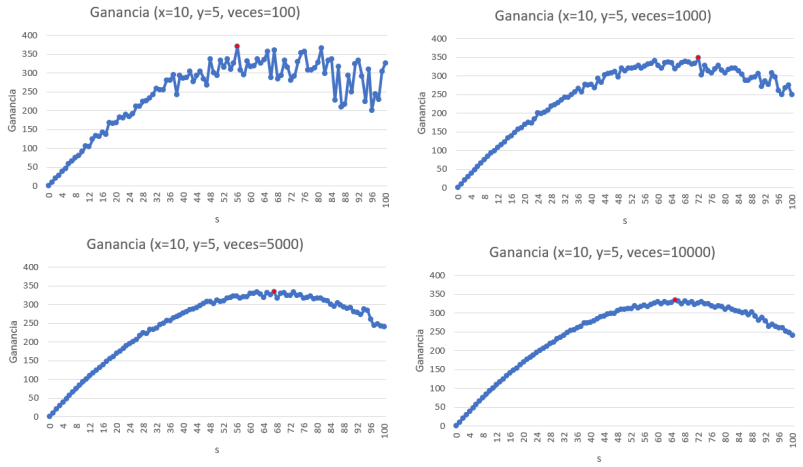

Figura 2: Evolución de la Ganancia (x=10, y=5) Distribución A

De igual forma ocurre en la Figura 3, donde el coste de perdida es de 10. El programa con esta distribución predice que durante un periodo de tiempo más largo el valor de ganancia decaerá proporcionalmente que aumenta el valor de 's'.

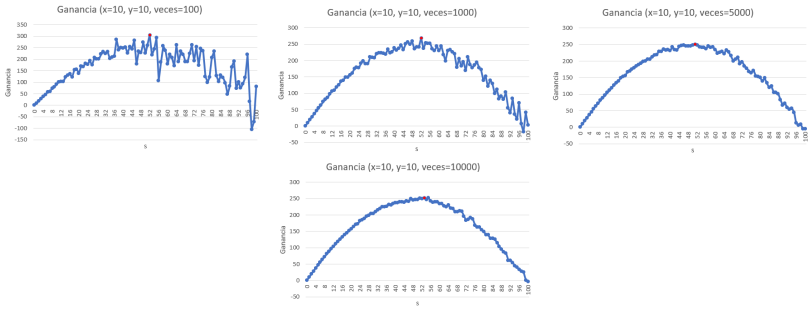

Figura 3: Evolución de la Ganancia (x=10, y=5) Distribución A

Respecto a las gráficas anteriores se puede observar en rojo el punto cuya ganancia es la mejor obtenida en todo el proceso.

x	y	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	1	100	93	505,62	292,842
		1000	95	469,872	314,4571
		5000	89	459,4292	308,1974
		10000	91	454,5472	310,8374
10	5	100	56	371,15	256,817
		1000	72	348,75	350,5631
		5000	67	348,75	350,5631
		10000	65	348,75	350,5631
10	10	100	52	348,75	350,5631
		1000	52	348,75	350,5631
		5000	51	348,75	350,5631
		10000	53	348,75	350,5631

Tabla 1: Resultados Distribución A

Por otra parte, en la Tabla 1 podemos ver los mejores resultados obtenidos por los diferentes parámetros sobre esta distribución. Como he comentado anteriormente se puede ver como el mejor valor de 's' obtenido es menor conforme el valor de perdida aumenta. Los mejores resultados de ganancia si comparamos entre los resultados obtenidos para los diferentes valores de 'y' son obtenidos para el valor de perdida 1, además la desviación en esta también es menor.

b) - Distribución B (Proporcional a 100-d)

En este apartado se ha utilizado una distribución proporcional a 100-d.

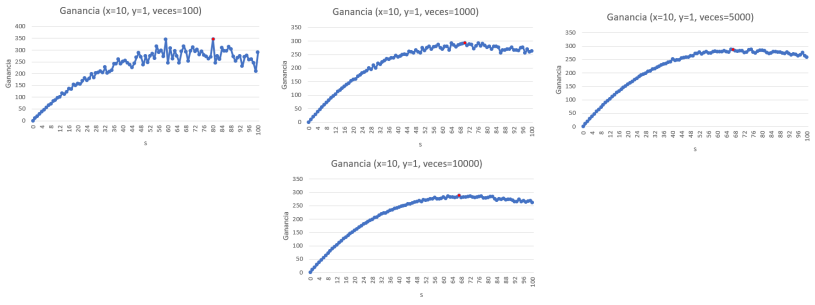

Figura 4: Evolución de la Ganancia (x=10, y=1) Distribución B

Como podemos ver la Figura 14 muestra la evolución de ganancia obtenida por cada valor 's' de las diferentes ejecuciones. Si observamos las gráficas donde el valor de 'veces' es superior a 1000 podemos ver como el valor de ganancia es creciente hasta que 's' tiene un valor próximo a 60, a partir de ese valor la ganancia se va equilibrando de forma moderada, también se puede apreciar como la ganancia empieza a decrecer al final de la ejecución. La gráfica con valor de veces=100 permite ver como los valores obtenidos presentan una gran desviación.

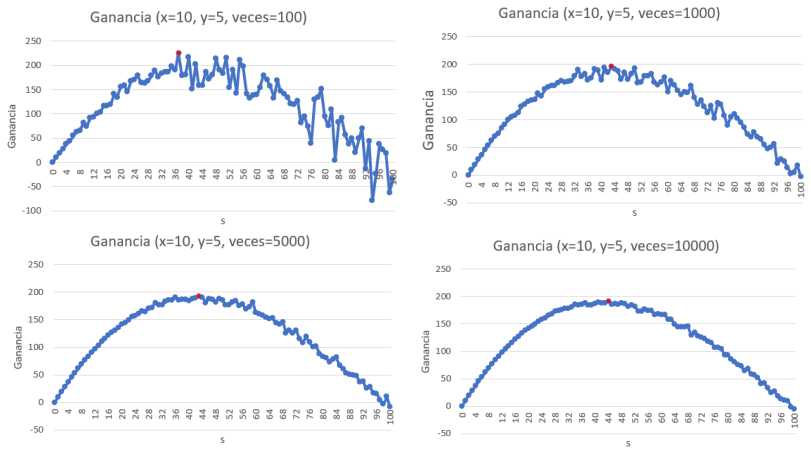

Figura 5: Evolución de la Ganancia (x=10, y=5) Distribución B

Mediante la observación de las gráficas obtenidas por esta distribución podemos decir que los valores son peores que los obtenidos en la distribución A. Vemos como los valores de 's' elevados causan una gran caída de la ganancia.

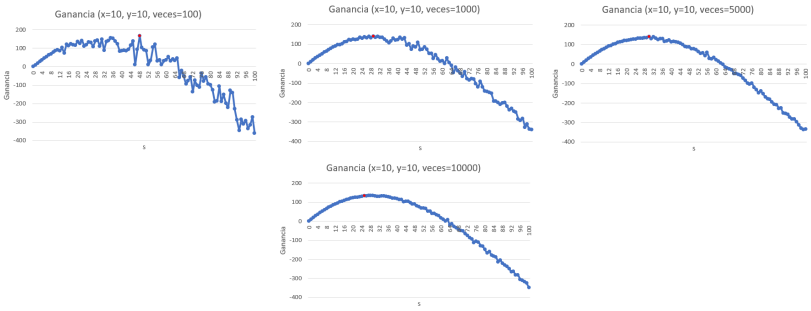

Figura 6: Evolución de la Ganancia (x=10, y=10) Distribución B

x	y	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	1	100	80	346,58	247,5504
		1000	70	293,396	245,138
		5000	67	287,7874	237,5177
		10000	67	288,5904	238,1739
10	5	100	37	226,3	177,2669
		1000	43	196,885	221,3657
		5000	43	192,586	226,7948
		10000	44	191,5025	231,2614
10	10	100	48	168,2	313,9207
		1000	29	142,68	192,9239
		5000	30	141,312	201,8927
		10000	25	135,572	162,8307

Tabla 2: Resultados Distribución B

Por otro lado, en la Tabla 2 vemos los mejores resultados obtenidos en las ejecuciones de la distribución B. Dos de las ejecuciones obtienen el valor s=67 como el mejor valor, las cuales son las ejecuciones con valor de veces=5000 y veces=100000, mientras que el resto de ejecuciones obtienen s=80 y s=70. Las desviaciones son similares entre los resultados. Si comparamos las ganancias, la mejor es obtenida por el valor veces=100 lo cual podría llevarnos a errores, ya que es mejor realizar cálculos para un largo periodo de tiempo ya que permitirá realizar un mejor balance sobre nuestras ganancias finales.

c) - Distribución C (Distribución "triangular")

En este apartado vamos a realizar los mismos experimentos que para las distribuciones anteriores. Esta distribución es triangular, es decir, para un rango de valores de 'd' la función de probabilidad es diferente al segundo rango de valores.

$$P(D=d) = \begin{cases} \frac{d}{2500} & si 0 \leq d < 50 \\ \frac{100-d}{2500} & si 50 \leq d \leq 99 \end{cases}$$

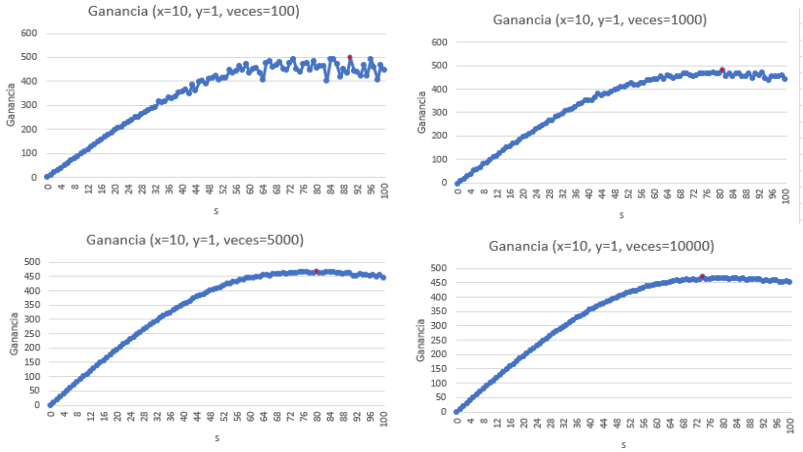

Figura 7: Evolución de la Ganancia (x=10, y=1) Distribución C

Mediante la Figura 7 vemos como el valor de ganancia crece considerablemente hasta llegar a un cierto valor de 's' donde a partir de ese valor la ganancia empieza a tomar valores equilibrados. Esto se debe a que llegamos a un cierto rango de 's' donde la ganancia experimenta poca variación entre esos valores.

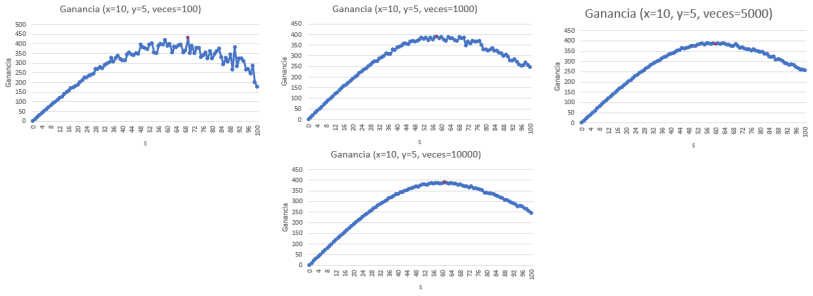

Figura 8: Evolución de la Ganancia (x=10, y=5) Distribución C

Cuando el valor de perdida es incrementado vemos como la ganancia decae cuando aumentados el valor de 's'. También se puede apreciar como el valor de veces ejerce una gran variación entre los valores de ganancia, es decir, cuando veces es pequeño vemos como la evolución de la ganancia tiene muchos picos altos y bajos, en cambio para valores de veces mayor estos picos disminuyen.

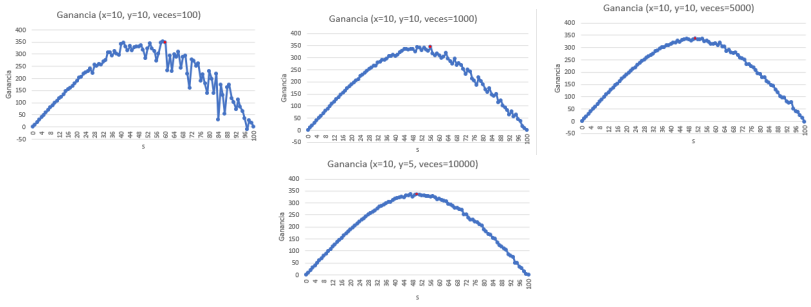

Figura 9: Evolución de la Ganancia (x=10, y=10) Distribución C

Para finalizar este apartado, vamos a analizar los resultados obtenidos en las diferentes ejecuciones. En la tabla 3 vemos los mejores resultados obtenidos en cada ejecución, como hemos visto en los resultados de las distribuciones anteriores cuando el valor de perdido es mayor el valor de 's' disminuye al igual que la ganancia, en cambio en esta distribución vemos como los valores de desviación son similares entre las ejecuciones de los diferentes parámetros.

x	y	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	1	100	90	494,32	225,3531
		1000	81	476,414	215,3639
		5000	80	466,3854	210,0757
		10000	74	467,7874	204,1891
10	5	100	69	429,15	264,5742
		1000	58	392,29	221,0417
		5000	59	387,965	222,0404
		10000	61	389,9275	227,0186
10	10	100	59	351,8	295,5713
		1000	56	343,78	263,3892
		5000	51	336,568	242,6352
		10000	50	335,688	235,6051

Tabla 3: Resultados Distribución C

1.2) - Modificaciones del Modelo

En este apartado vamos a realizar modificaciones sobre la manera de enfocar las perdidas cuando no se consiguen vender todos los productos.

1.2.1) Modificación 1 (costo de perdida fijo 'z')

La primera modificación consiste en fijar un valor 'z', el cual será la cantidad a pagar cuando los productos diarias no son vendidos por completo. Esto supone que por cada producto que no vendemos no paguemos ninguna cantidad de perdida, ya que para cualquier cantidad no vendida la perdida va a ser el valor 'z'.

Vamos a realizar pruebas de esta modificación sobre las distintas distribuciones y parámetros comentados en apartados anteriores. El valor de 'z' varia entre 10, 50 y 100 para todas las distribuciones.

-Distribución A-

x	z	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	10	100	90	550,70	2621,79
		1000	100	494,00	282,22
		5000	99	491,83	286,35
		10000	99	491,95	288,98
10	50	100	94	497,70	293,80
		1000	91	458,26	289,50
		5000	96	453,82	290,92
		10000	100	449,28	288,51
10	100	100	91	459,00	309,04
		1000	97	409,05	288,00
		5000	85	407,04	295,93
		10000	86	404,87	294,42

Tabla 4: Resultado Distribución A (Modificación 1)

Observando la Tabla 4, podemos ver como el valor de 's' obtenido como mejor es alto esto se debe a que la perdida por no vender todos los productos no es muy alta, lo cual permite pedir una mayor cantidad de productos ya que la ganancia aumentará mas rápido que perdidas se producen.

-Distribución B-

Esta distribución obtiene en media un valor de 's' más bajo que la primera distribución (aunque la diferencia no es elevada). Por consecuencia, el valor de ganancia obtenido es mas baja que la anterior esto se debe a que la frecuencia de perdida es mayor.

x	z	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	10	100	97	369,40	252,41
		1000	78	332,39	232,50
		5000	97	326,64	238,79
		10000	90	324,80	238,08
10	50	100	75	325,70	256,11
		1000	92	293,06	237,59
		5000	89	284,35	238,43
		10000	83	287,95	238,21
10	100	100	94	272,20	264,14
		1000	88	249,25	246,00
		5000	92	238,96	243,17
		10000	87	235,36	238,91

Tabla 5: Resultado Distribución B (Modificación 1)

-Distribución C-

Por último, observando la Tabla 6 vemos como los resultados en cuanto respecta al valor de 's' es muy parecido a la distribución B del apartado anterior, pero la ganancia es mucho mayor que la anterior.

x	z	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	10	100	93	520,80	197,75
		1000	80	504,10	196,08
		5000	91	496,92	201,94
		10000	86	492,63	199,95
10	50	100	75	482,00	190,28
		1000	86	466,67	210,70
		5000	85	455,90	205,73
		10000	94	454,48	204,37
10	100	100	92	458,30	217,26
		1000	88	409,48	212,08
		5000	80	409,98	214,00
		10000	78	406,63	211,81

Tabla 6: Resultado Distribución C (Modificación 1)

Conclusión

En conclusión, esta primera modificación obtiene mejores resultados que el programa original. Esto puede deberse al mejor rendimiento en la perdida de ganancia por los productos no vendidos. La ganancia conseguida por las diferentes ejecuciones mejora con respecto al original. Tenemos que destacar que no siempre será mejor esta opción ya que si todos los días no vendemos 3 productos y cada uno tiene una perdida de 1 euro, sería mas eficiente pagar 1 euro por cada producto que pagar una cuantía fija que puede ser mayor que esta cantidad, esto es lo más común en este tipo de contratos.

1.2.2) Modificación 2 (perdida $= x * d - \min\{z, (s-d) * y\}$)

En este experimento, tras la conclusión de la sección anterior, vamos a mejorar el rendimiento de las perdidas. Como hemos dicho antes no es lo mismo pagar 1 euro * 3 productos que una cuantía fija de 30 euros, por lo cual esta modificación consiste en seleccionar el mínimo valor entre dicha cuantía fija 'z' y el costo total de la suma de las perdidas de cada producto.

-Distribución A-

Comenzamos con las pruebas sobre la distribución A, como observamos en la Tabla 7 las diferencias en las ganancias mejoran un poco respecto a la modificación 1 y el programa original. La desviación es similar en ambas modificaciones.

x	y	z	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	1	10	100	92	540,54	259,77
			1000	93	502,55	286,81
			5000	94	491,89	286,53
			10000	95	487,44	288,91
10	5	50	100	95	513,05	326,85
			1000	86	461,83	286,85
			5000	94	457,71	294,42
			10000	94	452,60	294,56
10	10	100	100	79	458,10	295,17
			1000	95	416,62	296,94
			5000	88	412,65	299,72
			10000	85	408,41	300,69

Tabla 7: Resultado Distribución A (Modificación 2)

-Distribución B-

De igual forma pasa con la distribución B, se puede ver una leve mejoría sobre los resultados en comparación con la modificación 1.

x	y	z	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	1	10	100	82	368,97	230,37
			1000	84	336,06	234,60
			5000	96	325,47	238,71
			10000	97	322,88	238,06
10	5	50	100	74	318,60	246,15
			1000	84	295,62	243,08
			5000	68	284,20	228,14
			10000	75	289,31	237,35
10	10	100	100	75	294,40	254,93
			1000	55	252,54	224,92
			5000	68	237,06	241,61
			10000	85	237,48	244,61

Tabla 8: Resultado Distribución B (Modificación 2)

-Distribución C-

Por último, la distribución C sigue la misma armonía que las demás distribuciones, en ella se obtiene mejores resultados.

x	y	z	veces	Mejor_s	Mejor_ganancia	Mejor_desviacion
10	1	10	100	92	526,69	212,21
			1000	85	503,22	199,68
			5000	100	493,93	205,37
			10000	88	493,33	202,44
10	5	50	100	79	487,25	226,84
			1000	89	460,95	204,81
			5000	90	454,35	208,02
			10000	90	454,87	207,19
10	10	100	100	84	458,80	210,56
			1000	87	419,92	212,72
			5000	84	412,42	212,89
			10000	73	410,68	211,94

Tabla 9: Resultado Distribución C (Modificación 2)

Conclusión

Tras analizar los datos obtenidos en las ejecuciones de esta modificación podemos decir que esta mejora la ganancia final con respecto al programa original y la primera modificación. Si comparamos las diferentes distribuciones dentro de esta modificación vemos como la distribución A y C obtienen valores similares de ganancia, siendo estos valores superiores a la distribución B. Esta modificación es la más idónea desde el punto de vista del comerciante ya que las perdidas se adaptan a la menor cantidad.

Capitulo 2: Generadores de datos

2.1) Mejorando los generadores

En esta sección vamos a realizar mejoras en la eficiencia sobre los generadores utilizados en el capítulo anterior. La sección consta de 3 apartados los cuales contienen las diferentes mejoras. Estas mejoras son reordenamiento, búsqueda binaria y eliminaciones de operaciones explicitas en el caso A.

2.1.1) Mejora 1: Reordenamiento (caso C)

Mediante las tablas mostradas a continuación, podemos ver como el tiempo utilizado en la distribución C por medio del reordenamiento disminuye respecto al original, esto es debido a que cuando ordenamos la tabla en orden decreciente por medio de la probabilidad, cuando realizamos las comprobaciones dentro del bucle de la función genera-demanda() permitimos realizar la búsqueda de un menor valor al uniforme generado de forma mas rápida.

modificación	iteraciones	distribución	tiempo de ejecución (seg)
Original	10000	a	218,75
		b	140,62
		c	218,75
Reordenar C	10000	c	78,12

Tabla 10: Tiempos obtenidos con iteraciones=10000

modificación	iteraciones	distribución	tiempo de ejecución (seg)
Original	100000	a	21515,62
		b	1468,75
		c	7703,12
Reordenar C	100000	c	0,00

Tabla 11: Tiempos obtenidos con iteraciones=100000

Podemos ver como los tiempos obtenidos mediante esta mejora son mejores que la original. Esta mejora solo se a aplicado a la distribución C ya que las tablas de las demás distribuciones ya están ordenadas de forma decreciente según la probabilidad.

2.1.2) Mejora 2: Búsqueda binaria

Esta segunda mejora está basada en la búsqueda binaria. Esto consiste en buscar el dato entre un rango determinado de la tabla y en cada iteración que este no se encuentre el rango va disminuyendo, ya sea estrechando el rango por la derecha o por la izquierda de la tabla. Los resultados obtenidos son los siguientes:

modificación	iteraciones	distribución	tiempo de ejecución (seg)
Original	10000	a	218,75
		b	140,62
		c	218,75
Reordenar C	10000	c	78,12
R. Binaria	10000	a	0,00
		b	15,62
		c	0,00

Tabla 12: Tiempos obtenidos con iteraciones=10000

modificación	iteraciones	distribución	tiempo de ejecución (seg)
Original	100000	a	21515,62
		b	1468,75
		c	7703,12
Reordenar C	100000	c	0,00
R. Binaria	100000	a	46,88
		b	31,25
		c	31,25

Tabla 13: Tiempos obtenidos con iteraciones=100000

Analizando las tablas podemos ver como los tiempos obtenidos son mejores respecto al programa original y la mejora 1 (reordenamiento). Obteniendo para un valor de iteraciones = 10000 tiempos iguales a 0 seg.

2.1.3) Mejora 3: Evitar procesos explícitos (Caso A)

En esta mejora vamos a optimizar la función genera-demanda() para el caso A, esto se lleva a cabo mediante la eliminación de procesos explícitos en la generación de esta. El bucle interno de dicha función se sustituye por la operación $i = ceil((double)u * tama)$, esta operación es equivalente a realizar el bucle en el caso A ya que los valores están ordenados de manera decreciente según el valor de probabilidad (probabilidad constante), es decir, los valores (ordenados tanto por probabilidad como por sus propios valores) de la tabla también están ordenados de manera decreciente.

modificación	iteraciones	distribución	tiempo de ejecución (seg)
Original	100000	a	218,75
		b	140,62
		c	218,75
Reordenar C	100000	c	78,12
R. Binaria	100000	a	0,00
		b	15,62
		c	0,00
Evitar Op. Ex. (a)	100000	a	0,00

Tabla 14: Tiempos obtenidos con iteraciones=10000

modificación	iteraciones	distribución	tiempo de ejecución (seg)
Original	100000	a	21515,62
		b	1468,75
		c	7703,12
Reordenar C	100000	c	0,00
R. Binaria	100000	a	46,88
		b	31,25
		c	31,25
Evitar Op. Ex. (a)	100000	a	0,00

Tabla 15: Tiempos obtenidos con iteraciones=100000

En conclusión, observando las tablas 14 y 15 vemos como el tiempo empleado por este generador es igual a 0, por lo que esta mejora es la más óptima en lo que respecta al caso A.

2.2) Generadores congruenciales

En este apartado se ha implementado diferentes formas de calcular las ecuaciones congruenciales descritas en el guión de prácticas.

Métodos:

- **Aritmética entera:** realiza el calculo directo mediante el módulo.
- **Aritmética real artesanal:** calculamos un primer valor mediante la división de (a*x+c) entre el valor m, a este valor le restamos su parte entera y multiplicamos por m.
- **Aritmética real artesanal corregida:** realizamos los mismos calculos que el anterior pero al resultado obtenido le sumamos 0.5 y casteamos a entero lo que permitirá corregir los errores producidos mediante los redondeos de los cálculos previos.
- **Aritmética real con fmod:** este método devuelve el resto de dividir x entre y.

xn+1 = (2061xn + 4321) mod m	Periodo
Arit. Entera	10000
Arit. Real Artesanal	130
Arit. Real Artesanal Corregida	227
Arit. Real fmod	10000

xn+1 = (2060xn + 4321) mod m	Periodo
Arit. Entera	6
Arit. Real Artesanal	5
Arit. Real Artesanal Corregida	6
Arit. Real fmod	6

Tabla 16: Periodos obtenidos por los métodos

En conclusión, tras analizar la Tabla 16 donde nos muestra los periodos obtenidos por los métodos descritos anteriormente, podemos decir que respecto a la primera ecuación congruacional los métodos más óptimos son el cálculo mediante aritmética entera y aritmética real fmod ya que en estos se consiguen un periodo mayor. Además, en los cálculos de aritmética de punto fijo no ocurren perdidas por redondeos en cambió las operaciones que contienen aritmética de punto flotante pueden llevar a errores por redondeo. Por lo que en los métodos de aritméticas real artesanal y artesanal corregida donde he utilizado variables de tipo float obtenemos un periodo menor, esto puede deberse a lo comentado anteriormente de las perdidas por redondeos.

Por otro lado, en lo que respecta a la segunda ecuación congruacional, vemos como los periodos obtenidos por los diferentes métodos son menores que los anteriores, esto se debe a que al modificar el valor 'a' (2060), después de pocos calculos reiterados por medio de la ecuación obtenemos siempre el mismo valor, lo cual provoca que el programa finalice mucho antes.