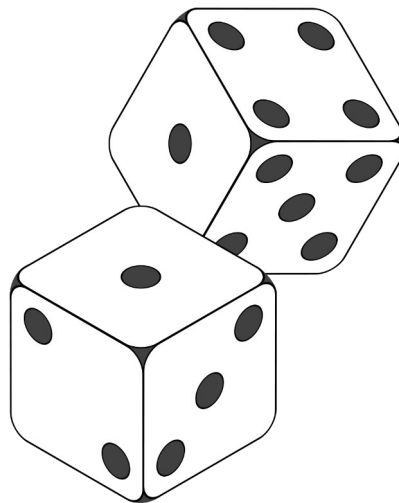




ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ
“ΓΕΝΝΗΤΡΙΕΣ ΤΥΧΑΙΩΝ ΑΡΙΘΜΩΝ”



ΠΡΟΠΤΥΧΙΑΚΟΣ ΦΟΙΤΗΤΗΣ: ΠΛΑΤΑΝΙΑΣ ΘΕΟΔΩΡΟΣ ΑΜ Π2011002
ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΑΥΛΩΝΙΤΗΣ ΜΑΡΚΟΣ

ΚΕΡΚΥΡΑ ΙΑΝΟΥΑΡΙΟΣ 2021

Ευχαριστίες

Ολοκληρώνοντας τη παρούσα πτυχιακή εργασία με τίτλο “Γεννήτριες τυχαίων αριθμών” , θα ήθελα να ευχαριστήσω όλους όσους μου συμπαράστανται σε αυτή τη προσπάθεια. Αρχικά οφείλω να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Αυλωνίτη Μάρκο για την καθοδήγηση και υποστήριξη που μου πρόσφερε όχι μόνο κατά τη διάρκεια της παρούσας εργασίας αλλά καθ’ όλη την φοιτητική μου πορεία. Επίσης θα ήθελα να εκφράσω τις ευχαριστίες μου σε όλο το διοικητικό και εκπαιδευτικό προσωπικό που συνέβαλαν στην απόκτηση των απαραίτητων γνώσεων για την επιτυχή φοίτηση μου. Τέλος οφείλω να ευχαριστήσω την οικογένειά μου για τη αμείωτη συμπαράσταση και στήριξη ως προς την επίτευξη των στόχων μου.

Περίληψη

Η παρούσα πτυχιακή εργασία έχει ως σκοπό την κατηγοριοποίηση, μελέτη και ανάλυση των σημαντικότερων μεθόδων παραγωγής τυχαίων αριθμών. Στο πρώτο μέρος, παρουσιάστηκε ο ορισμός της τυχειότητας, μία ιστορική αναδρομή καθώς επίσης και τα είδη γεννητριών με τις σύγχρονες εφαρμογές τους. Στη συνέχεια αναλύθηκαν οι ιδιότητες και τα χαρακτηριστικά των πιο διαδεδομένων γεννητριών ψευδοτυχειότητας. Επιπλέον παρουσιάστηκε ο αλγόριθμος Mersenne Twister και ο τρόπος που κατασκευάζεται η πιο γνωστή γεννήτρια MT19937. Τέλος η εργασία περιλαμβάνει το σχεδιασμό και την υλοποίηση μιας εφαρμογής slot machine μαζί με την υλοποίηση της γεννήτριας MT19937 που χρησιμοποιεί.

Abstract

The purpose of this thesis is the study, analysis and the categorization of the most important methods which generate random numbers. A definition of randomness, a historical flashback and the categories of basic random generators were presented in the first part. Next there is an analysis about the properties and characteristics of the most know pseudorandom number generators. In addition, the algorithm Mersenne Twister and the way the most famous generator MT19937 is made were presented. In the end, this thesis includes the design and the implementation of the a slot machine application together with the implementation of MT19937 generator.

Πρόλογος

Αντικείμενο της παρούσας πτυχιακής εργασίας αποτέλεσε η ανάλυση των γεννητριών τυχαίων αριθμών (Random Number Generators – RNGs) που χρησιμοποιούνται καθημερινά σε μεγάλο εύρος εφαρμογών. Τα πιο χαρακτηριστικά παραδείγματα ανήκουν στον κλάδο της κρυπτογραφίας και ασφάλειας δικτύων, της στατιστικής και μαθηματικών, των προσομοιώσεων καθώς επίσης εφαρμογή βρίσκουν στα τυχερά παιχνίδια. Κύριος σκοπός της εργασίας είναι η θεωρητική μελέτη των ψευδοτυχαίων γεννητριών και των ντετερμινιστικών μεθόδων που χρησιμοποιούν για να παράγουν τυχαίους αριθμούς.

Αναλυτικότερα στο Κεφάλαιο 1, αναφέρεται ο ορισμός της τυχειότητας και παρουσιάζονται συνοπτικά τα είδη των γεννητριών ανάλογα με την διαδικασία που χρησιμοποιούν για να παράγουν τυχαίους αριθμούς. Επίσης αναφέρεται μία σύντομη ιστορική αναδρομή της τυχειότητας και οι τομείς που έχουν εφαρμογή σήμερα.

Στο Κεφάλαιο 2 γίνεται ανάλυση των σπουδαιότερων ντετερμινιστικών μεθόδων που χρησιμοποιούν οι ψευδοτυχαίες γεννήτριες καθώς και τις ιδιότητες και χαρακτηριστικά τους. Επιπλέον γίνεται αναφορά στην αριθμητική με modulus την οποία κάνουν χρήση οι περισσότεροι μέθοδοι.

Στην συνέχεια στο Κεφάλαιο 3 παρουσιάζεται ο αλγόριθμος Mersenne Twister που αποτελεί μία από τις πιο χρησιμοποιούμενες γεννήτριες σήμερα. Επίσης δίνεται η διαδικασία που ακολουθεί ο αλγόριθμος σε φυσική γλώσσα και ψευδοκώδικα.

Τέλος στο Κεφάλαιο 4 γίνεται ο σχεδιασμός και η υλοποίηση σε Python ενός τυχαίου παιχνιδιού το οποίο εφαρμόζει την παραλλαγή της γεννήτριας Mersenne Twister MT19937.

Περιεχόμενα

1 Εισαγωγή.....	10
1.1 Η αξία της τυχαιότητας.....	10
1.2 Ο ορισμός της έννοιας της τυχαιότητας.....	11
1.3 Ιστορική αναδρομή.....	12
1.4 Κατηγοριοποίηση γεννητριών.....	14
1.4.1 Πραγματικές γεννήτριες τυχαίων αριθμών.....	15
1.4.2 Ψευδοτυχαίες γεννήτριες τυχαίων αριθμών.....	17
1.4.3 Σύγκριση γεννητριών.....	19
1.5 Σύγχρονες εφαρμογές.....	20
2 Μέθοδοι παραγωγής ψευδοτυχαίων αριθμών.....	23
2.1 Σπόρος και περίοδος.....	23
2.2 Αριθμητική με modulus.....	24
2.3 Γραμμικές γεννήτριες ισοδυναμίας.....	29
2.4 Πολλαπλές αναδρομικές γεννήτριες.....	35
2.5 Γεννήτρια Fibonacci με καθυστέρηση	37
2.6 Γεννήτριες ισοδυναμιών με πίνακες.....	39
2.7 Μη γραμμικές γεννήτριες.....	41
2.7.1 Αντίστροφες γεννήτριες ισοδυναμιών.....	42
2.7.2 Επιπλέον μη γραμμικές γεννήτριες.....	43
2.8 Linear feedback shift register γεννήτριες.....	45
2.9 Generalized feedback shift register γεννήτριες.....	48
2.10 Συνδυαστικές γεννήτριες.....	50
3 Mersenne Twister.....	51
3.1 Η γεννήτρια Mersenne twister.....	51
3.2 Αλγοριθμική ανάλυση.....	52
3.3 Παρουσίαση του Mersenne twister με φυσική γλώσσα.....	55

3.4	Παράμετροι και περιοδικότητα.....	56
3.5	Παρουσίαση του Mersenne twister με ψευδοκώδικα.....	57
4	Εφαρμογή slot machine.....	59
4.1	Εισαγωγή	59
4.2	Κανόνες και χαρακτηριστικά παιχνιδιού.....	61
4.3	Περιγραφή εφαρμογής slot machine.....	66
4.4	Συνδυασμοί και πιθανότητες.....	68
4.5	Υλοποίηση εφαρμογής.....	70
4.5.1	Η γλώσσα Python.....	71
4.5.2	Υλοποίηση Mersenne twister.....	74
4.5.3	Υλοποίηση κλάσης slot machine.....	75
4.5.4	Υλοποίηση διεπαφής.....	76
5	Κατάλογος Σχημάτων / Πινάκων.....	79
6	Βιβλιογραφία.....	80

Κεφάλαιο 1^ο

ΕΙΣΑΓΩΓΗ

1.1 Η Αξία Της Τυχειότητας

Η τυχειότητα αποτελεί θεμέλιο στοιχείο του κόσμου. Αν κάποιος το αμφισβητεί αρκεί να αναλογιστεί πως ακόμα και η ίδια του η ύπαρξη οφείλεται σ' ένα τυχαίο γεγονός. Υπάρχουν αμέτρητα τέτοια παραδείγματα που συμβαίνουν "τυχαία" στην καθημερινότητα και αποδεικνύουν το καθοριστικό ρόλο του "τυχαίου" στην ζωή. Άλλωστε αν δεν υπήρχε, όλα θα ήταν αυστηρώς καθορισμένα και θα είχε χαθεί το νόημα της ύπαρξης και της εξέλιξης του ανθρώπου.

Η ύπαρξη της τυχειότητας έχει γίνει αντιληπτή εδώ και αρκετές χιλιετίες. Στη προσπάθεια τους να βάλουν τάξη στην ανεξήγητη ροή των γεγονότων, οι άνθρωποι δημιούργησαν την έννοια της "μοίρας". Αργότερα επινόησαν τις πιθανότητες. Η θεωρία των πιθανοτήτων αρχικά δημιουργήθηκε για χάρη των τυχαίων παιχνιδιών, μέχρι οι επιστήμονες να αντιληφθούν την πραγματική δύναμη της και τη χρησιμοποιήσουν σε άλλα επιστημονικά ζητήματα. Σήμερα λόγω της ραγδαίας εξέλιξης της τεχνολογίας και των επιστημών, η χρήση τυχαίων αριθμών σε διάφορες εφαρμογές και επιστημονικές έρευνες έχει αυξηθεί σε μεγάλο βαθμό.

Όπως γίνεται αντιληπτό η αξία της τυχειότητας και των τυχαίων αριθμών είναι τεράστια στην καθημερινή ζωή του ανθρώπου.

1.2 Ορισμός Της Έννοιας της Τυχειότητας

Η ερώτηση σχετικά με το τι είναι πραγματικά τυχαίο οδηγεί τους ερευνητές συχνά σε φιλοσοφικές και στοχαστικές συζητήσεις. Αυτό συμβαίνει λόγω της διαφορετικής ερμηνείας που μπορεί να δώσει κάποιος στον όρο "τυχαίος". Η πιο ουσιαστική προσέγγιση για την πληροφορική έγινε με την ανάπτυξη της θεωρίας των πληροφοριών από το Claude Shannon και την εισαγωγή της έννοιας της εντροπίας. Η θεωρία που διατύπωσε ήταν ότι η τυχειότητα είναι αντίθετη από τον ντετερμινισμό σε μία στοχαστική διαδικασία. Έτσι εάν ένα σύστημα έχει εντροπία μηδέν δεν έχει τυχειότητα ενώ η αύξηση της εντροπίας οδηγεί στην αύξηση και αυτής. Οι Solomonov και Kolmogorov ακολούθησαν μία διαφορετική πτυχή στο θέμα της τυχειότητας, που είχε να κάνει με το πως προσεγγίζεται η ύπαρξη του τυχαίου. Ο Kolmogorov όρισε πως μία ακολουθία ονομάζεται τυχαία, αν το μήκος της είναι ίσο με το μήκος του συντομότερου προγράμματος που χρειάζεται για να παραχθεί.

Πλέον αντίθετα από την οπτική του Kolmogorov, μελετάτε η αλληλεπίδραση με το περιβάλλον και όχι η οντότητα και η κατασκευή της.

Συγκεκριμένα για να πιστοποιηθεί μία ακολουθία ότι είναι τυχαία θα πρέπει να πληρεί τις εξής ιδιότητες.

Ομοιόμορφη κατανομή : Η κατανομή των αριθμών στην ακολουθία πρέπει να είναι ομοιόμορφη στο διάστημα από το οποίο προήλθαν, δηλαδή η συχνότητα εμφάνισης κάθε αριθμού θα πρέπει να είναι κατά προσέγγιση η ίδια .

Ανεξαρτησία : Καμία τιμή της ακολουθίας δεν μπορεί να προκύψει από τις άλλες.

1.3 Ιστορική Αναδρομή

Όπως μπορεί κάποιος να διαπιστώσει από την προηγούμενη παράγραφο, υπάρχουν διαφορετικοί ορισμοί για την έννοια της τυχαιότητας, κάποιοι είναι πιο αφηρημένοι και άλλοι πιο αυστηρά καθορισμένοι. Από την αρχαιότητα έχουν δοθεί διάφορες ερμηνείες για το τι είναι πραγματικά τυχαίο. Αρχικά η έννοια του τυχαίου και της τυχαιότητας συσχετιζόταν με αυτόν της μοίρας. Ο πρώτος λαός που ασχολήθηκε με τις πιθανότητες και την τύχη ήταν οι Κινέζοι 3000 χρόνια π.Χ. . Στην συνέχεια Έλληνες φιλόσοφοι όπως ο Δημόκριτος, ο Αριστοτέλης έδωσαν τις δικές τους ερμηνείες. Για αιώνες η τύχη διερευνόταν χωρίς μαθηματικό υπόβαθρο, μέχρι τον 16ο αιώνα, όταν Ιταλοί μαθηματικοί άρχισαν να τυποποιούν τις πιθανότητες που σχετίζονταν με τυχερά παιχνίδια.

Πρώτη γνωστή απόπειρα θεώρησης της τυχαιότητας έγινε από τον Leibniz τον 17ο αιώνα ο οποίος αναρωτώμενος αν ένα σύνολο σημείων πάνω στο χαρτί επιλέχθηκαν τυχαία η όχι, κατέληξε στο συμπέρασμα ότι είναι τυχαία αν η μαθηματική εξίσωση που τα περιέγραφε ήταν πολύπλοκη σε μεγάλο βαθμό. Αυτή η θεωρία ονομάστηκε τρεις αιώνες αργότερα ως αλγοριθμική τυχαιότητα από τον Kolmogorov. Η Θεωρία των πιθανοτήτων αναπτύχθηκε ακόμα περισσότερο το 18ο αιώνα με τις αξιοσημείωτες εργασίες των μαθηματικών Bernoulli, De Moivre, Laplace και Gauss. Ο Laplace δεν περιορίζεται μόνο στην μαθηματική ανάλυση των τυχερών παιχνιδιών, αλλά εφαρμόζει το συμπέρασμα τους σε ένα πλήθος από επιστημονικά και πρακτικά προβλήματα. Έτσι με αφορμή τη μελέτη των σφαλμάτων που προκύπτουν στις επαναλαμβανόμενες μετρήσεις του ίδιου

αστρονομικού μεγέθους ανακαλύπτεται η περίφημη κατανομή του Gauss. Αργότερα η προσέγγιση των πιθανοτήτων από άποψη συχνότητας αναπτύχθηκε για πρώτη φορά από τον Robert Ellis και τον John Venn στα τέλη του 19ου αιώνα με την έκδοση του βιβλίου Η Λογική της Τύχης (The Logic of Chance) το 1880 όπου παρουσιάστηκαν αποδείξεις για την τυχαιότητα των ψηφίων του αριθμού π .

Κατά την διάρκεια του 20ου αιώνα έγινε σημαντική πρόοδος στην μελέτη και κατανόηση των εννοιών της τυχαιότητας και των πιθανοτήτων. Ο Richard von Mises στη μελέτη της θεωρίας συχνοτήτων προχώρησε το έργο του John Venn συσχετίζοντας την "αταξία" ή τυχαιότητα ενός δείγματος με την έλλειψη επιτυχιών σε τυχερά παιχνίδια. Η ανάπτυξη της θεωρίας των πληροφοριών από τον Claude Shannon το 1948 αποτέλεσε την αρχή για την θεώρηση της τυχαιότητας από την σκοπιά της εντροπίας. Στα τέλη της δεκαετίας του 1960 μελετήθηκαν για πρώτη φορά οι τυχαίες ακολουθίες από τους Kolmogorov και Solomonov, οι οποίοι διατύπωσαν την θεωρία ότι η αλγοριθμική τυχαιότητα μίας ακολουθίας καθορίζονταν ως το ελάχιστο μέγεθος ενός προγράμματος που εκτελούμενο σε ένα υπολογιστή παράγει την ακολουθία αυτή. Περίπου δέκα χρόνια αργότερα το 1970, οι επιστήμονες συνειδητοποίησαν ότι η εισαγωγή της τυχαιότητας σε υπολογισμούς οδηγεί στο σχεδιασμό και δημιουργία καλύτερων αλγορίθμων.

Επίσης τον 20ο αιώνα γεννήθηκε η ανάγκη για χρήση τυχαίων αριθμών σε επιστημονικές εργασίες. Ο μόνος εφικτός τρόπος εκείνη την περίοδο ήταν να παραχθούν τυχαίοι αριθμοί από κάποιες τυχαίες διαδικασίες υπό τις κατάλληλες συνθήκες όπως η ρίψη ενός ή περισσότερων ζαριών και η καταγραφή των αποτελεσμάτων. Η πρώτη

οργανωμένη συλλογή τυχαίων ψηφίων έγινε από τον L.H.C.Tippett το 1927 ο οποίος κατέγραψε σ' ένα αρχείο 40.000 ψηφία. Από εκείνη την στιγμή, δημιουργήθηκαν ποικίλες διαδικασίες οι οποίες είχαν ως σκοπό να παράγουν τυχαίου αριθμούς στηριζόμενες σε μαθηματικές αρχές. Το 1939, οι M.G.Kendall και B.Badington – Smith δημιούργησαν την δική τους ευρεσιτεχνία για να παράγουν ένα πίνακα από 100.000 τυχαία ψηφία, ενώ το 1955 η RAND Corporation δημοσίευσε ένα πίνακα που αποτελούνταν από εκατομμύρια. Η χρήση των πινάκων και γραπτών εγγράφων, παρόλο που θεωρούνταν ορθή, υστερούσε σε μεγάλο βαθμό λόγω της περιορισμένης χωρητικότητας και του χρόνου που απαιτούνταν για την δημιουργία τους. Αυτά τα μειονεκτήματα οδήγησαν τους επιστήμονες στην αυτοματοποίηση των διαδικασιών με την χρήση του υπολογιστή , ακόμα και στην πρώιμη μορφή του την δεκαετία του 1940. Το 1949 ο John von Neumann ανέπτυξε μία αριθμητική μέθοδο εκμεταλλευόμενος τις λειτουργίες και δυνατότητες των υπολογιστών εκείνης της εποχής, που ονόμασε "Middle – Square" μέθοδο. Η βασική ιδέα είναι να υπολογίζει το τετράγωνο του προηγούμενου τυχαίου αριθμού κάθε φορά και να εξάγει τα μεσαία ψηφία. Μέχρι σήμερα δημιουργούνται ή βελτιώνονται διαδικασίες παραγωγής τυχαίων αριθμών.

1.4 Κατηγοριοποίηση Γεννητριών

Με την έλευση των υπολογιστών, διαπιστώθηκε η ανάγκη για την εισαγωγή της τυχειότητας σε ένα πρόγραμμα αυτού ώστε να χρησιμοποιείται στις πολυάριθμες εφαρμογές που υπάρχουν και αυξάνονται καθημερινά. Ωστόσο αυτό δεν μπορεί να επιτευχθεί εύκολα.

Ένας υπολογιστής ακολουθεί εντολές με ένα τρόπο που είναι απολύτως προβλέψιμος και αυτό καθιστά δύσκολο τον προγραμματισμό του να κάνει κάτι τυχαίο. Υπάρχουν δύο κύριες προσεγγίσεις για να παραχθεί ένα τυχαίος αριθμός με την χρήση ενός υπολογιστή.

Η πρώτη προσέγγιση λαμβάνει ως είσοδο τυχαία γεγονότα από τον πραγματικό κόσμο για να δημιουργήσει στην έξοδο μία ακολουθία τυχαίων αριθμών. Αυτή η προσέγγιση αναφέρεται ως πραγματική γεννήτρια τυχαίων αριθμών, γιατί σε κανονικές συνθήκες είναι αδύνατο για οποιονδήποτε να προβλέψει τον επόμενο αριθμό της ακολουθίας. Η δεύτερη προσέγγιση θεωρεί ότι οι αλγόριθμοι με απρόβλεπτα αποτελέσματα επαρκούν για την ικανοποίηση των απαιτήσεων της τυχαιότητας. Αυτές οι γεννήτριες που παράγουν ακολουθίες μέσω διάφορων αλγοριθμικών τεχνικών ονομάζονται ψευδοτυχαίες γεννήτριες τυχαίων αριθμών. Ονομάζονται κατά αυτό τον τρόπο διότι κάθε τιμή προσδιορίζεται με βάση την κατάσταση του συστήματος και δεν είναι πραγματικά τυχαία.

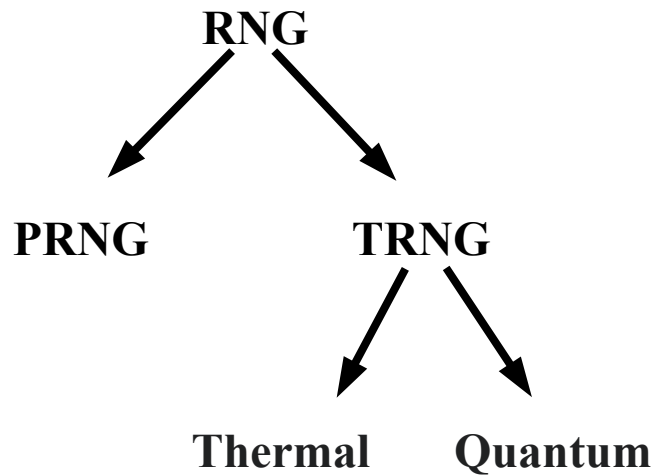
1.4.1 Πραγματικές Γεννήτριες Τυχαίων Αριθμών (True Random Number Generators - TRNGs)

Όπως αναφέρθηκε παραπάνω μια πραγματική γεννήτρια τυχαίων αριθμών εισάγει τα φυσικά φαινόμενα που υπάρχουν στον πραγματικό κόσμο στον υπολογιστή και παράγει στην έξοδο πραγματικά τυχαίους, ανεξάρτητους και απρόβλεπτους αριθμούς. Τέτοια φυσικά φαινόμενα μπορεί να είναι το στρίψιμο ενός νομίσματος ή η ρίψη ενός ζαριού.

Ωστόσο η χρήση τέτοιων φαινομένων θα περιόριζαν σε μεγάλο βαθμό την παραγωγή τυχαίων αριθμών. Η χαμηλή αποδοτικότητα είναι ένα από τα κύρια προβλήματα που αντιμετωπίζουν οι περισσότερες πραγματικά τυχαίες γεννήτριες.

Τα πιο συχνά φαινόμενα που χρησιμοποιούνται ως είσοδο σ' αυτές τις γεννήτριες είναι τα θερμικά φαινόμενα λόγω της εύκολης ανίχνευσης τους. Τέτοια μπορεί να είναι ο θερμικός θόρυβος μίας αντίστασης, ο θόρυβος χιονοστιβάδας (avalanche noise) από μία αντίστροφα πολωμένη ένωση $p - n$ ή ο ατμοσφαιρικός θόρυβος. Μία άλλη κατηγορία είναι τα κβαντικά φαινόμενα όπως είναι η ραδιενεργός διάσπαση ή η κίνηση των φωτονίων. Η χρήση τέτοιων φαινομένων που ανήκουν σε μία από τις δύο κατηγορίες παρόλο που έχουν υψηλό ποσοστό παραγωγής τυχαίων αριθμών, απαιτούν πολύ περισσότερο χρόνο για να εξάγουν ένα αποτέλεσμα. Επίσης η χρήση τέτοιων φαινομένων καθιστούν τις πραγματικές γεννήτριες μη ντετερμινιστικές που σημαίνει ότι δεν μπορούν να έχουν την ίδια ακολουθία αριθμών ως αποτέλεσμα, παρ' όλα αυτά μία ακολουθία μπορεί να προκύψει ξανά τυχαία. Αυτό σημαίνει ότι οι γεννήτριες που ανήκουν σε αυτή την κατηγορία είναι μη περιοδικές.

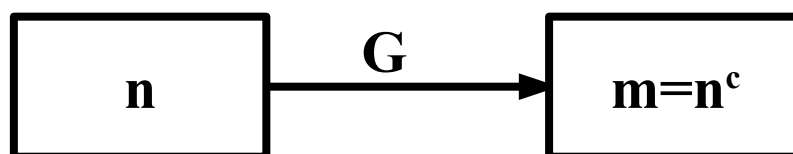
Από την στιγμή που οι πραγματικές γεννήτριες απαιτούν την χρήση συμβάντων στον πραγματικό κόσμο για την λειτουργία τους προκύπτει ένα κύριο αρνητικό, η αναγκαία χρήση κάποιου υλικού για την ανίχνευση τους. Αυτό μπορεί να κάνει τέτοιες τύπου γεννήτριες ακριβές, ειδικά αν το υλικό που απαιτείται δεν είναι συνηθισμένο. Η ανάγκη κάποιου υλικού για την υλοποίηση τους σημαίνει επίσης ότι είναι ευάλωτες σε φυσικές επιθέσεις ή αν δεν υπάρχουν τέτοιες παραμένουν ευάλωτες στο χρόνο και σε σφάλματα στην κατασκευή του υλικού.



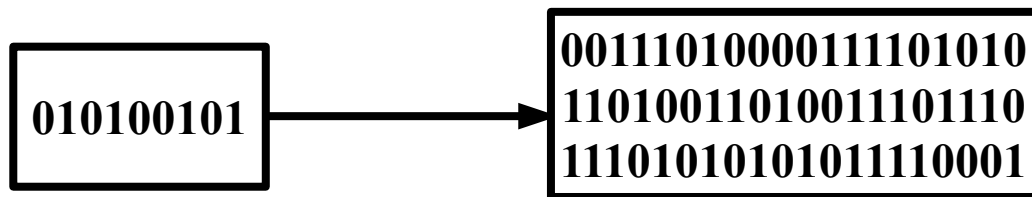
Σχήμα 1.1: Αναπαράσταση της ταξινόμησης των RNGs

1.4.2 Ψευδοτυχαίες Γεννήτριες Τυχαίων Αριθμών (Pseudo Random Number Generators – PRNGs)

Αντίθετα με τις πραγματικά τυχαίες γεννήτριες τυχαίων αριθμών οι ψευδοτυχαίες γεννήτριες στηρίζονται στην ύπαρξη ενός ή περισσότερων ντετερμινιστικών αλγορίθμων, για την παραγωγή των νέων κάθε φορά αριθμών. Θα μπορούσαμε να πούμε ότι είναι κάποιες συναρτήσεις που παράγουν ακολουθίες από bits, ξεκινώντας από μία μικρή τυχαία ακολουθία από bits για να αρχίσει η μετάβαση στην επόμενη τιμή. Αυτή η μικρή ακολουθία ονομάζεται σπόρος (seed).



Σχήμα 1.2: Παράδειγμα Γεννήτριας Ψευδοτυχειότητας



Σχήμα 1.3: Παράδειγμα Γεννήτριας Ψευδοτυχαιότητας με τυχαίο σπόρο

Ένας γενικός ορισμός για τις PRNGs είναι :

Ορισμός : Μία γεννήτρια παραγωγής ψευδοτυχαίων αριθμών είναι ένας αλγόριθμος G , πολυωνυμικού χρόνου, ο οποίος λαμβάνει ως είσοδο n (το πλήθος bits) τιμές και παράγει ως έξοδο m bits με $m > n$.

Η μαθηματική έκφραση που χρησιμοποιούν οι περισσότερες γεννήτριες ψευδοτυχαιότητας είναι της μορφής

$$x_i = f(x_{i-1}, x_{i-2}, \dots, x_{i-k})$$

Οι προηγούμενοι k αριθμοί καθορίζουν τον επόμενο αριθμό. Στην πράξη το πιο σύνηθες είναι, ο αριθμός που παράγεται να είναι συνάρτηση μόνο του αμέσως προηγούμενου. Στην πλειονότητα οι αριθμοί που παράγονται παρουσιάζουν ομοιόμορφη κατανομή σε ένα διάστημα $[a,b]$. Ένα παράδειγμα είναι οι γραμμικές γεννήτριες ισοδυναμίας που θα αναλύσουμε στο επόμενο κεφάλαιο.

Στόχος αυτής της κατηγορίας γεννητριών είναι να προσεγγίσουν με ένα , φτηνό και γρήγορο τρόπο τις ιδιότητες των πραγματικών γεννητριών. Οι παραγόμενες ακολουθίες φαινομενικά, έχουν όλα τα απαραίτητα

χαρακτηριστικά για να καλούνται τυχαίες, αποκρύπτοντας το γεγονός ότι έχουν δημιουργηθεί με ένα ντετερμινιστικό τρόπο. Η ντετερμινιστική φύση τους όμως, τις αναγκάζει να παράγουν τις ίδιες ακολουθίες αν εισαχθεί ο ίδιος σπόρος.

1.4.3 Σύγκριση Γεννητριών

Τα χαρακτηριστικά των γεννητριών ψευδοτυχαίων αριθμών (PRNGs) και των γεννητριών πραγματικά τυχαίων αριθμών (TRNGs) διαφέρουν τόσο μεταξύ τους που δεν μπορεί κάποιος να πει ότι η κατηγορία γεννητριών υπερτερεί έναντι της άλλης. Η επιλογή μιας γεννήτριας γίνεται ανάλογα με τις απαιτήσεις της εφαρμογής στην οποία θα χρησιμοποιηθεί, κάποια από τις δύο κατηγορίες είναι η πιο κατάλληλη για αυτή ώστε να υπάρχει το επιθυμητό αποτέλεσμα.

Οι γεννήτριες ψευδοτυχαίων αριθμών έχουν υψηλή αποδοτικότητα, το οποίο σημαίνει ότι μπορούν να παράγουν πολλούς αριθμούς που είναι φαινομενικά τυχαίοι σε μικρό χρονικό διάστημα. Επιπλέον η δημιουργία των αριθμών γίνεται με ντετερμινιστικό τρόπο, δηλαδή μία ακολουθία μπορεί να παραχθεί ξανά αργότερα αν ο σπόρος είναι γνωστός. Αυτό δεν μπορεί να επιτευχθεί με τις πραγματικές γεννήτριες παρά μόνο από σύμπτωση καθώς είναι μη ντετερμινιστικές.

Αυτά τα χαρακτηριστικά κάνουν τις ψευδοτυχαίες γεννήτριες κατάλληλες για προσομοιώσεις και εφαρμογές μοντελοποίησης οι οποίες απαιτούν πιθανόν μεγάλο όγκο δεδομένων ενώ οι μη ντετερμινιστικές πραγματικές γεννήτριες χρησιμοποιούνται συχνά σε εφαρμογές όπως η κρυπτογράφηση

ή τα παιχνίδια.

Μία άλλη διαφορά ανάμεσα στις δύο κατηγορίες βρίσκεται στην περιοδικότητα. Οι ψευδοτυχαίες γεννήτριες σε αντίθεση με τις πραγματικές είναι περιοδικές, που σημαίνει ότι στο τέλος η ακολουθία αριθμών που δημιουργήθηκε θα επαναληφθεί από την αρχή. Τέλος ένα τελευταίο χαρακτηριστικό που ξεχωρίζει τις δύο κατηγορίες μεταξύ τους είναι το κόστος. Οι πραγματικές γεννήτριες έχουν υψηλό κόστος λόγω του υλικού που απαιτείται για την κατασκευή τους .

Χαρακτηριστικά	PRNG	TRNG
Αποδοτικότητα	Υψηλή	Χαμηλή
Ντετερμινισμός	Ντετερμινιστική	Μη ντετερμινιστική
Περιοδικότητα	Περιοδική	Απεριοδική
Κόστος	Υψηλό	Χαμηλό

Πίνακας 1.1: Χαρακτηριστικά των PRNGs και TRNGs.

1.5 Σύγχρονες Εφαρμογές

Ο λόγος που χρησιμοποιούνται αυτές διάφορες γεννήτριες, δεν είναι άλλος από τον μεγάλο εύρος εφαρμογών που απαιτούν πλέον ένα βαθμό τυχαιότητας είτε μικρό είτε μεγαλύτερο και πιθανόν καθοριστικό ώστε να είναι επιτυχής ο σκοπός της κάθε μίας. Η τυχαιότητα έχει πολλές χρήσεις στις επιστήμες, στατιστική, στην ασφάλεια και κρυπτογραφία, στα τυχερά παιχνίδια ακόμη και στην τέχνη και πολιτική.

Στο πεδίο της στατιστικής πολλά στοιχεία βασίζονται στους τυχαίους

αριθμούς και κατ επέκταση στην τυχαιότητα. Ένα από αυτά μπορεί να είναι η επιλογή ενός αντιπροσωπευτικού δείγματος από ένα σύνολο ή η ανάλυση και περιγραφή δεδομένων που είναι αποτελέσματα επιστημονικών πειραμάτων. Σε πολλά επιστημονικά πεδία εκτός από την ανάλυση αποτελεσμάτων χρησιμοποιούνται προσομοιώσεις στον υπολογιστή για να αναπαραχθούν και να μελετηθούν πραγματικά φαινόμενα. Ένα σύνολο τέτοιων προσομοιώσεων βασίζονται σε τυχαίους αριθμούς και χρησιμοποιούν συνήθως ψευδοτυχαίες γεννήτριες παραγωγής αριθμών για την δημιουργία τους.

Οι απρόβλεπτοι αριθμοί επίσης παίζουν σημαντικό ρόλο σε ποικίλες εφαρμογές στο κλάδο της κρυπτογραφίας και ασφάλειας δικτύων ώστε να παρέχεται όσο μεγαλύτερη ασφάλεια στις σύγχρονες επικοινωνίες. Τέτοιες εφαρμογές για παράδειγμα μπορεί να είναι οι κινητές επικοινωνίες, πληρωμές μέσω διαδικτύου, ATM, το e-banking, ασύρματα κλειδιά και άλλες. Όπως γίνεται αντιληπτό η ασφάλεια είναι απαραίτητη σε τέτοιες εφαρμογές, για αυτό το λόγο χρησιμοποιούνται κρυπτογραφικοί αλγόριθμοι των οποίων το κλειδί επιλέγεται με υψηλή τυχαιότητα ώστε οι επιθέσεις σε τέτοια συστήματα να είναι όσο το δυνατόν πιο δύσκολη. Οι απαιτήσεις αυτές καθιστούν τις γεννήτριες πραγματικά τυχαίων αριθμών ιδανικές και όταν χρήση αυτών δεν είναι εφικτή χρησιμοποιούνται υψηλού επιπέδου γεννήτριες ψευδοτυχειότητας.

Όσον αφορά τα τυχερά παιχνίδια υπάρχει η εντύπωση ότι το ενδιαφέρον των ανθρώπων για αυτά αποτελεί το κύριο κίνητρο για την ανάπτυξη θεωριών τυχαιότητας. Πολλές συσκευές τυχαιοποίησης όπως το ζάρι, ανακατέματος καρτών, η ρουλέτα αναπτύχθηκαν για να χρησιμοποιηθούν σε παιχνίδια τύχης. Στα σύγχρονα καζίνο υπάρχουν μία ή

περισσότερες γεννήτριες παραγωγής τυχαίων αριθμών που αποφασίζουν το αποτέλεσμα σε ένα παιχνίδι. Στις σύγχρονες μηχανές slot η θέση που θα σταματήσουν οι τροχοί έχει αποφασιστεί ήδη από το λογισμικό την στιγμή που πατήθηκε το κουμπί της έναρξης.

Στην τέχνη υπάρχει ισχυρισμός ότι αισθητικές θεωρίες βασίζονται στην τυχειότητα. Ένα παράδειγμα είναι η τοποθέτηση αντικειμένων τέχνης με τυχαίο τρόπο σε μία έκθεση. Άλλοι καλλιτέχνες χρησιμοποιούν τεχνικές συσκευές τυχειότητας ως κομμάτι της δουλειάς τους όπως ο Ellsworth Kelly. Στη μουσική περιλαμβάνεται στη στοχαστική, αλλειωτική ή γεννητική μουσική. Τέλος στην πολιτεία χρησιμοποιούνται τεχνικές τυχειότητας ώστε να γίνει η επιλογή ενόρκων στα νομικά συστήματα όπως συμβαίνει στην Αγγλία και στις Ηνωμένες Πολιτείες της Αμερικής.

Κεφάλαιο 2^ο

ΜΕΘΟΔΟΙ ΠΑΡΑΓΩΓΗΣ ΨΕΥΔΟΤΥΧΑΙΩΝ ΑΡΙΘΜΩΝ

2.1 Σπόρος Και Περίοδος

Οι περισσότερες κατηγορίες ψευδοτυχαίων αριθμών αριθμών στηρίζονται σε μία αναδρομική σχέση ώστε να δημιουργήσουν κάθε φορά ένα τυχαίο αριθμό στην έξοδο. Όπως είδαμε στο προηγούμενο κεφάλαιο η απλούστερη σχέση που απεικονίζει αυτή την αναδρομή είναι η :

$$x_i = f(x_{i-1}, x_{i-2}, \dots, x_{i-k}) \quad (1)$$

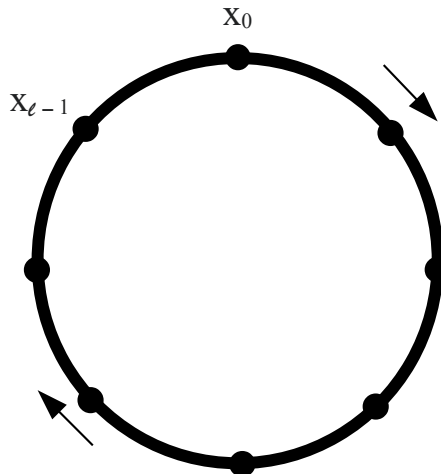
Σε κάθε επανάληψη το πλήθος των αριθμών που απαιτούνται για να δημιουργηθεί ο νέος αριθμός ισοδυναμεί με το k το οποίο καλείται τάξη της γεννήτριας. Χρησιμοποιώντας λοιπόν μία αναδρομική σχέση και δεδομένου ότι αυτή παράγει αριθμούς σε ένα πεπερασμένο σύνολο, είναι αναπόφευκτο ότι κάποια χρονική στιγμή η γεννήτρια θα εμφανίσει στην έξοδο ξανά κάποιο αριθμό που έχει εμφανιστεί και προηγουμένως. Το μήκος της ακολουθίας αριθμών που παράγεται από τη κάθε γεννήτρια χωρίς να περιλαμβάνει κάποιον επαναλαμβανόμενο αριθμό ονομάζεται περίοδος ή κύκλος της γεννήτριας.

Όπως είναι λογικό η αναδρομική σχέση θα πρέπει να ξεκινήσει από

κάποια αρχική τιμή ή τιμές. Το σύνολο των τιμών που δέχεται η κάθε γεννήτρια ως είσοδο για να αρχίσει η διαδικασία παραγωγής ονομάζεται σπόρος (seed). Λόγω της ντετερμινιστικής φύσης των αλγορίθμων που χρησιμοποιεί κάθε ψευδοτυχαία γεννήτρια όταν εισαχθεί ο ίδιος σπόρος σε κάποια από αυτές θα παράγει την ίδια ακολουθία αριθμών κάθε φορά.

Συμπερασματικά, μία γεννήτρια ξεκινώντας από ένα σπόρο με τιμή x_0 και δημιουργώντας ℓ διαφορετικές τιμές στην έξοδο θα καταλήξει στον όρο $x_{\ell-1}$, ο οποίος αποτελεί τον τελευταίο πριν εμφανιστεί ξανά το x_0 .

Στο παρακάτω σχήμα φαίνεται αυτή η διαδικασία.



Σχήμα 2.1: Αναπαράσταση κύκλου μιας PRNG

2.2 Αριθμητική Με Modulus

Οι περισσότερες ψευδοτυχαίες γεννήτριες παραγωγής αριθμών, όπως θα δούμε στις επόμενες παραγράφους, χρησιμοποιούν τον τελεστή modulo (mod) σε διάφορες υπολογιστικές σχέσεις.

Για αυστηρά θετικό ακέραιο m και $a, b \in \mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$ λεμέ ότι ο a είναι ισοδύναμος με το b modulo m αν ο m διαιρεί το $a - b$ ($m \mid (a - b)$). Αυτή η σχέση διατυπώνεται ως :

$$a \equiv b \pmod{m} \quad (2)$$

Εδώ να επισημάνουμε ότι ένας ακέραιος b διαιρεί έναν ακέραιο a , αν υπάρχει $c \in \mathbb{Z}$ τέτοιο ώστε να ισχύει $a = b \cdot c$. Η σχέση a καλείται σχέση ισοδυναμίας ή απλά ισοδυναμία (congruence). Οι ακέραιοι a, b λέμε ότι είναι ισοδύναμοι (congruent) modulo m , εάν $(a \bmod m) = (b \bmod m)$. Ο αριθμός m που υπάρχει ονομάζεται διαιρέτης (modulus) της ισοδυναμίας.

ΘΕΩΡΗΜΑ: Έστω m ένας θετικός ακέραιος. Για κάθε ακέραιο a , υπάρχει ένας μοναδικός ακέραιος b , τέτοιος ώστε $a \equiv b \pmod{m}$ για $0 \leq b < m$.

Ακολουθούν συνοπτικά οι ιδιότητες της Σχέσης (2) για m θετικός ακέραιος και $a, b, c \in \mathbb{Z}$.

1. $a \equiv a \pmod{m}$
2. $a \equiv b \pmod{m}$ συνεπάγεται $b \equiv a \pmod{m}$
3. $a \equiv b \pmod{m}$ και $b \equiv c \pmod{m}$ συνεπάγεται $a \equiv c \pmod{m}$

Ένα χαρακτηριστικό παράδειγμα χρήσης του τελεστή modulo στους υπολογιστές είναι η πράξη $x \bmod 2$ στον προγραμματισμό. Για κάθε ακέραιο x η πράξη δίνει αποτέλεσμα 1 ή 0 καθώς όλοι οι ακέραιοι αριθμοί μπορούν να γραφούν είτε στην μορφή $x = 2k + 1$ είτε στη μορφή $x = 2k$,

για $k \in \mathbb{Z}$.

Από την παραπάνω ανάλυση προκύπτει ότι ο τελεστής modulo αντιστοιχίζει όλους τους ακεραίους στο σύνολο $\{0, 1, 2, \dots, (n - 1)\}$. Στο ερώτημα αν είναι εφικτό να πραγματοποιηθούν αριθμητικές πράξεις με modulo εντός του συνόλου αυτό η απάντηση είναι καταφατική. Οι πράξεις αυτές αποτελούν την Αριθμητική με Modulus ή αλλιώς Modular Αριθμητική.

Η αριθμητική με modulus χαρακτηρίζεται από τις εξής ιδιότητες :

1. $(a \bmod m) + (b \bmod m) = (a + b) \bmod m$
2. $(a \bmod m) - (b \bmod m) = (a - b) \bmod m$
3. $(a \bmod m) \cdot (b \bmod m) = (a \cdot b) \bmod m$

Η ύψωση σε δύναμη πραγματοποιείται με επαναλαμβανόμενο πολλαπλασιασμό όπως και στην κανονική αριθμητική.

Ορίζουμε το σύνολο $\mathbb{Z}_n$ ως το σύνολο των μη αρνητικών ακεραίων μικρότερων του n :

$$\mathbb{Z}_n = \{0, 1, \dots, (n - 1)\}$$

Αυτό είναι γνωστό ως σύνολο υπολοίπων ή κλάσεις υπολοίπου (residue classes) . Κάθε ακέραιος του $\mathbb{Z}_n$ αποτελεί αντιπρόσωπο της κλάσης. Μία κλάση υπολοίπου συμβολίζεται με $[a]$ ή $[a]_n$.

Παρακάτω φαίνονται οι κλάσεις υπολοίπου modulo 4 ως παράδειγμα.

$$[0] = \{ \dots, -16, -12, -8, -4, 0, 4, 8, 12, 16, \dots \}$$

$$[1] = \{ \dots, -15, -11, -7, -3, 1, 5, 9, 13, 17, \dots \}$$

$$[2] = \{ \dots, -14, -10, -6, -2, 2, 6, 10, 14, 18, \dots \}$$

$$[3] = \{ \dots, -13, -9, -5, -1, 3, 7, 11, 15, 19, \dots \}$$

Εάν πραγματοποιήσουμε πράξεις με modulus στο Z_n , ισχύουν οι παρακάτω ιδιότητες για κάθε κλάση υπολοίπου.

1. Αντιμεταθετική Ιδιότητα

$$(a + b) \bmod m = (b + a) \bmod m$$

$$(a \cdot b) \bmod m = (b \cdot a) \bmod m$$

2. Προσεταιριστική Ιδιότητα

$$[(a + b) + c] \bmod m = [a + (b + c)] \bmod m$$

$$[(a \cdot b) \cdot c] \bmod m = [a \cdot (b \cdot c)] \bmod m$$

3. Επιμεριστική Ιδιότητα

$$[a \cdot (b + c)] \bmod m = [(a \cdot b) + (a \cdot c)] \bmod m$$

4. Ταυτόσημα Στοιχεία

$$(0 + a) = a \bmod m$$

$$(1 \cdot a) = a \bmod m$$

5. Προσθετικός Αντίστροφος ($-a$)

Για κάθε $a \in Z_n$, υπάρχει ένα z τέτοιο ώστε

$$a + z \equiv 0 \bmod m$$

6. Πολλαπλασιαστικός Αντίστροφος (a^{-1})

Για κάθε $a \in Z_n$, υπάρχει ένα z τέτοιο ώστε

$$a \cdot z \equiv 1 \bmod m$$

Όπως φαίνεται η αριθμητική με modulus στον μεγαλύτερο βαθμό είναι ίδια με την κανονική αριθμητική. Ωστόσο υπάρχει μια ιδιαιτερότητα που την ξεχωρίζει. Για να ισχύει

$$(a \cdot b) \equiv (a \cdot c) \pmod{m} \text{ τότε } b \equiv c \pmod{m} \quad (3)$$

θα πρέπει οι ακέραιοι a, m να είναι πρώτοι μεταξύ τους (relatively prime). Δύο ακέραιοι καλούνται πρώτοι μεταξύ τους αν ο μόνος θετικός κοινός τους διαιρέτης είναι το 1. Ο λόγος είναι ότι, όταν κάνουμε πράξεις modulo m για οποιοδήποτε m , ένα συντελεστής a που πολλαπλασιάζει διαδοχικά τους ακέραιους $(0, 1, \dots, m - 1)$ θα αποτύχει να δημιουργήσει ένα πλήρες σύνολο, αν τα a και m έχουν κάποιους κοινούς διαιρέτες.

Στις γεννήτριες παραγωγής τυχαίων αριθμών, σημαντική σημασία έχουν οι κλάσεις υπολοίπων των οποίων τα μέλη είναι σχετικά πρώτοι αριθμοί με το modulus m . Το πλήθος αυτών των κλάσεων ισούται με το πλήθος των ακεραίων μεταξύ 0 και $m - 1$ οι οποίοι είναι σχετικώς πρώτοι με το m και συμβολίζεται με το $\phi(m)$. Γενικότερα ισχύει για πρώτο αριθμό m , ισχύει $\phi(m) = m - 1$. Η συνάρτηση $\phi(m)$ ονομάζεται συνάρτηση ϕ του Euler και κατέχει σημαντικό ρόλο στον υπολογισμό περιόδων των ψευδοτυχαίων γεννητριών.

Μία σημαντική ιδιότητα για την συνάρτηση του Euler είναι ότι για πρώτο αριθμό p και θετικό ακέραιο a ισχύει

$$\phi(p^a) = p^{a-1}(p-1) \quad (4)$$

Επίσης ένα θεώρημα που έχει μεγάλη χρησιμότητα στην δημιουργία ψευδοτυχαίων γεννητριών είναι το Θεώρημα Fermat το οποίο λέει

Έστω n θετικός ακέραιος που είναι σχετικώς πρώτος με ένα $a \in \mathbb{Z}$. Θεωρούμε την ακολουθία των δυνάμεων του a που ανήκουν στο $\mathbb{Z}_n^*$. Το $\mathbb{Z}^*$ συμβολίζει το σύνολο όλων των κλάσεων υπολοίπου που έχουν πολλαπλασιαστικό αντίστροφο

$$[1]_n = a^0, a^1, a^2, \dots$$

Αφού κάθε τέτοια δύναμη είναι ένα στοιχείο του $\mathbb{Z}_n^*$ και αφού το $\mathbb{Z}_n^*$ είναι πεπερασμένο σύνολο, αυτή η ακολουθία δυνάμεων πρέπει να αρχίσει να επαναλαμβάνεται σε κάποιο σημείο. Δηλαδή θα πρέπει να υπάρχει θετικός ακέραιος k τέτοιος ώστε $a^k = a^i$ για κάποιο $i = 0, \dots, k-1$ με $a^0, a^1, \dots, a^{k-1}$ διακεκριμένες δυνάμεις. Αυτή η τιμή k ονομάζεται πολλαπλασιαστική τάξη του a ή πολλαπλασιαστική τάξη $a \bmod n$ και μπορεί να χαρακτηριστεί ως ο μικρότερος θετικός ακέραιος k τέτοιος ώστε

$$a^k \equiv 1 \pmod{n} \quad (5)$$

Στη συνέχεια θα δούμε την εφαρμογή των ιδιοτήτων και των θεωρημάτων σε διάφορες κατηγορίες ψευδοτυχαίων γεννητριών και τους λόγους που αποτελούν σημαντικά εργαλεία για την δημιουργία αυτών

2.3 Γραμμικές Γεννήτριες Ισοδυναμίας (Linear Congruential Generator – LCG)

Ο D. H. Lehmer το 1951 πρότεινε μία απλή γεννήτρια ισοδυναμίας που αποτέλεσε την πιο απλή μορφή ψευδοτυχειότητας. Σε αυτή, ο κάθε αριθμός παράγεται μέσω μίας απλής γραμμικής συνάρτησης, στην έξοδο της

οποίας με τη χρήση του τελεστή modulo εφαρμόζεται μία αναγωγή σ' ένα συγκεκριμένο πεδίο τιμών.

Η ακολουθία των τυχαίων αριθμών x_n αποκτάται μέσω της ακόλουθης γενικής αναδρομικής εξίσωσης :

$$x_n = (\alpha x_{n-1} + c) \bmod m \quad (6)$$

όπου m ο διαιρέτης (modulo) της ισοδυναμίας, α ο πολλαπλασιαστής (multiplier), c ο προσθετικός παράγοντας (increment) ενώ το x_0 αποτελεί τον σπόρο (seed). Για κάθε μία από τις μεταβλητές ισχύουν οι περιορισμοί $m > 0$, $0 < \alpha < m$ και $0 \leq x_0 < m$.

Όταν $c = 0$ η εξίσωση (6) γίνεται

$$x_n = (\alpha x_{n-1}) \bmod m, \quad 0 \leq x_n < m \quad (7)$$

και η γεννήτρια που βασίζεται σε αυτή την εξίσωση ονομάζεται Πολλαπλασιαστική Γεννήτρια Ισοδυναμίας (Multiplicative Congruential Generator – MCG) ενώ για $c \neq 0$ αποκαλείται Μεικτή (Mixed Linear Congruential Generator – MLCG). Μία ακολουθία που προκύπτει από την Σχέση (7) καλείται ακολουθία Lehmer. Κάθε τιμή των συναρτήσεων x_n αντιστοιχίζεται στο μοναδιαίο διάστημα $(0, 1)$, αν το διαιρέσουμε με το m . Η διαδικασία αυτή ονομάζεται κανονικοποίηση.

$$U_n = \frac{x_n}{m} \quad (8)$$

Η επιλογή των τιμών για τα α , c , x_0 και m είναι κρίσιμη για την κατασκευή μίας καλής LCG. Αν η επιλογή αυτή γίνει σωστά τότε οι

ακολουθίες που παράγονται θα μοιάζουν με αυτές που δημιουργούνται από μία πραγματικά τυχαία γεννήτρια και θα παρουσιάζουν μία ομοιόμορφα κατανομημένη συμπεριφορά στο διάστημα μεταξύ του 0 και του 1. Ας επιλέξουμε για παράδειγμα τις τιμές $\alpha = 7$, $c = 0$, $m = 32$ και $x_0 = 1$. Τότε η Σχέση (6) γίνεται

$$x_n = (7x_{n-1}) \bmod 32$$

και οι αριθμοί που παράγονται από αυτή έπειτα από 10 επαναλήψεις είναι οι εξής :

n	x_{n-1}	αx_{n-1}	x_n	x_n/m
0	—	—	1	—
1	1	7	7	0.218
2	7	49	17	0.531
3	17	119	23	0.718
4	23	147	1	0.031
5	1	7	7	0.218
6	7	49	17	0.531
7	17	119	23	0.718
8	23	147	1	0.031
9	1	7	7	0.218
10	7	49	17	0.531

Πίνακας 2.1: Αποτελέσματα δέκα επαναλήψεων της $x_n = (7x_{n-1}) \bmod 32$

Παρατηρώντας την ακολουθία που παράγεται διαπιστώνει κανείς ότι μετά από 4 επαναλήψεις εμφανίζονται οι ίδιοι αριθμοί στην έξοδο από την

αρχή, δηλαδή η περίοδος της σχέσης είναι 4. Είναι προφανές ότι δεν είναι ικανοποιητική. Ας δούμε τώρα την ακολουθία που δημιουργείται αν αλλάξουμε την τιμή του a σε 5. Η Σχέση (6) μετατρέπεται σε

$$x_n = (5x_{n-1}) \bmod 32$$

και τα αποτελέσματα μετά από 10 επαναλήψεις είναι :

n	x_{n-1}	ax_{n-1}	x_n	x_n/m
0	—	—	1	—
1	1	5	5	0.156
2	5	25	25	0.781
3	25	125	29	0.906
4	29	145	17	0.531
5	17	85	21	0.656
6	21	105	9	0.281
7	9	45	13	0.406
8	13	65	1	0.031
9	1	5	5	0.156
10	5	25	25	0.781

Πίνακας 2.2: Αποτελέσματα δέκα επαναλήψεων της $x_n = (5x_{n-1}) \bmod 32$

Όπως φαίνεται στο παραπάνω Πίνακα η περίοδος της γεννήτριας αυξήθηκε σε 8.

Επειδή το x_n καθορίζεται από το x_{n-1} , και αφού υπάρχουν m διαφορετικές τιμές για το x , η μέγιστη περίοδος μιας LCG ισούται με m . Για την περίπτωση που $c = 0$, η μέγιστη περίοδος είναι ίση με $m - 1$, διότι

το x_{n-1} δεν μπορεί να είναι ποτέ 0. Ιδανικά θα θέλαμε να παράγεται μία μακριά ακολουθία διακεκριμένων τυχαίων αριθμών και για αυτό το λόγο επιλέγεται ένα πολύ μεγάλο m . Ένα σύνηθες κριτήριο είναι το m να παίρνει την τιμή κοντά ή ίση με τον μέγιστο θετικό ακέραιο που μπορεί να αναπαρασταθεί από έναν υπολογιστή. Άρα επιλέγεται ως τιμή κάποια κοντά ή ίση με το 2^{31} . Μία τέτοια γεννήτρια που χρησιμοποιεί αυτά τα χαρακτηριστικά είναι η RANDU η οποία αναπτύχθηκε από την IBM στη δεκαετία του 1960. Είναι μία απλή γραμμική γεννήτρια ισοδυναμίας που χρησιμοποιεί την αναδρομική σχέση $x_n = (2^{16} + 3) \bmod 2^{31}$.

Πλέον οι διαιρέτες m που χρησιμοποιούνται στις LCGs με $c = 0$ είναι πρώτοι Mersenne. Ένας πρώτος Mersenne καλείται ο αριθμός που είναι της μορφής $2^p - 1$. Ένας ευρέως διαδεδομένος διαιρέτης είναι ο $2^{31} - 1$ που χρησιμοποιείται στα 32 – bit υπολογιστικά συστήματα.

Επομένως η Σχέση (1) γίνεται :

$$x_n = (\alpha x_{n-1}) \bmod (2^{31} - 1) \quad (9)$$

Υπάρχουν περισσότερες από 2 δισεκατομμύρια πιθανές τιμές για το α , από τις οποίες λίγες μόνο είναι ικανοποιητικές. Μία από αυτές είναι $\alpha = 7^5 = 16807$, η οποία χρησιμοποιήθηκε αρχικά στους υπολογιστές IBM 360. Έτσι αντικαθιστώντας το α στην παραπάνω σχέση έχουμε

$$x_n = (7^5 x_{n-1}) \bmod (2^{31} - 1) \quad (10)$$

Αυτή η LCG έχει αναλυθεί σε μεγάλο βαθμό όσο καμία άλλη γεννήτρια και προσφέρεται ιδιαίτερα για εφαρμογές στατιστικής και προσομοίωσης.

Στη γενική περίπτωση, η περίοδος μίας MCG εξαρτάται από τη μικρότερη θετική τιμή του k που ικανοποιεί την εξής ισοδυναμία

$$a^k = 1 \bmod m \quad (11)$$

Μετά η ακολουθία αρχίζει να επαναλαμβάνεται. Επομένως, η περίοδος δεν μπορεί να ξεπεράσει την τιμή του k . Επιπρόσθετα, εάν ισχύει ότι a και m είναι σχετικά πρώτοι μεταξύ τους τότε ισχύει σύμφωνα με το θεώρημα Euler – Fermat ότι

$$a^{\varphi(m)} = 1 \bmod m \quad (12)$$

Το μήκος περιόδου δεν μπορεί να είναι μεγαλύτερο από $\varphi(m)$. Επομένως αναζητάμε μία τιμή a τέτοια ώστε το k της Σχέσης (11) να ισούται με το $\varphi(m)$. Αυτές οι τιμές που ικανοποιούν την παραπάνω Σχέση (12) ονομάζονται αρχικές ρίζες modulo m .

Όπως αναφέραμε η ισχύς των LCGs προκύπτει από την επιλογή του πολλαπλασιαστή και του modulo. Αν αυτή η επιλογή είναι σωστή τότε μία ακολουθία που θα προκύψει θα είναι στατιστικά αξεχώριστη από οποιαδήποτε ακολουθία επιλέγεται τυχαία με τιμές στο σύνολο $(1, 2, \dots, m-1)$. Ωστόσο εκτός από την επιλογή του σπόρου x_0 που επιλέγεται τυχαία, όλοι οι άλλοι αριθμοί της ακολουθίας έπονται ντετερμινιστικά. Αυτό δημιουργεί κινδύνους.

Έστω ότι κάποιος επιτιθέμενος γνωρίζει ότι χρησιμοποιείται ο αλγόριθμος γραμμικών ισοδυναμιών και επιπλέον είναι γνωστές οι παράμετροι τότε μόλις ανακαλυφθεί ένας και μόνο αριθμός, θα γίνουν

γνωστοί αυτομάτως και όλοι οι διαδοχικοί του. Ακόμη και εάν ο επιτιθέμενος γνωρίζει μόνο τη χρήση του αλγορίθμου, τότε με τη γνώση ενός μικρού τμήματος της ακολουθίας που παράγεται μπορεί να προσδιορίσει τις παραμέτρους. Ας θεωρήσουμε ότι μπόρεσε να ανακαλύψει τις τιμές των x_1, x_2, x_3 . Τότε λύνοντας το σύστημα εξισώσεων ως προς a, c και m

$$x_1 = (ax_0 + c) \bmod m$$

$$x_2 = (ax_1 + c) \bmod m$$

$$x_3 = (ax_2 + c) \bmod m$$

μπορεί να προσδιορίσει τις τιμές αυτών.

Ο κλάδος της κρυπτογραφίας και της κρυπτανάλυσης προσφέρει διάφορους τρόπους και τεχνικές ώστε να αποφευχθούν τέτοιου είδους κίνδυνοι που δεν θα αναλυθούν στην παρούσα διπλωματική.

2.4 Πολλαπλές Αναδρομικές Γεννήτριες (Multiple Recursive Generators – MRG)

Οι ανάγκες της σημερινής εποχής καθιστούν τις LCG μη επαρκής για κάποιες εφαρμογές λόγω της μικρής περιοδικότητας που παρουσιάζουν. Για την αντιμετώπιση αυτού του προβλήματος χρησιμοποιούνται άλλοι τρόποι παραγωγής τυχαίων αριθμών που είναι αποδοτικοί και εύκολοι να εφαρμοστούν. Πολλοί τέτοιοι τρόποι βασίζονται

στην χρήση διανύσματος μήκους μεγαλύτερου της μονάδας στη θέση του k που είδαμε μέχρι τώρα

$$x_n = (a^T U_{n-1} + c) \bmod m, 0 \leq x_i \leq m \quad (13)$$

Όπου το a συμβολίζει ένα διάνυσμα σταθερών μήκους k , ενώ το U_{n-1} αντιστοιχεί σε ένα επίσης διάνυσμα το οποίο αποτελείται από k στοιχεία $(x_{n-1}, x_{n-2}, \dots, x_{n-k})$. Όπως είναι λογικό θα απαιτείται και για σπόρος διάνυσμα k τιμών. Το k καλείται τάξη της γεννήτριας.

Για $c = 0$ η Σχέση (12) μετατρέπεται σε :

$$x_n = (a_1 x_{n-1} + a_2 x_{n-2} + \dots + a_k x_{n-k}) \bmod m \quad (14)$$

Η γεννήτρια που βασίζεται στην παραπάνω αναδρομική σχέση ονομάζεται Πολλαπλή Αναδρομική γεννήτρια (MRG) ή Πολλαπλή Αναδρομική Πολλαπλασιαστική Γεννήτρια Ισοδυναμίας (Multiple Recursive Multiplicative Congruential Generator) για $k > 1$. Ενώ για $k = 1$ συμπίπτει με την Πολλαπλασιαστική Γεννήτρια Ισοδυναμίας (MLCG).

Οι MRGs χωρίζονται σε δύο κατηγορίες σύμφωνα με την τιμή του m . Η πρώτη βασίζεται στο $m = 2$ για να δημιουργηθεί ακολουθία από bits. Αυτή η κατηγορία είναι η βάση για διάφορες γεννήτριες τυχαίων αριθμών όπως είναι οι Shift Register, οι Generalized Feedback Shift Register συμπεριλαμβανομένου και της Mersenne – Twister "MT19937".

Οι γεννήτριες της δεύτερης κατηγορίας κατασκευάζονται με μεγάλο modulus για να πετύχουν μεγάλη περίοδο. Συνήθως το m είναι πρώτος αριθμός και παίρνει την τιμή $2^{31}-1$. Επιλέγεται επίσης μεγάλη τιμή k . Με

αυτές τις επιλογές των m και k η περίοδος μπορεί να φτάσει την τιμή m^k-1 που είναι πολύ μεγαλύτερη σε σχέση με αυτή των LCGs.

Όσον αναφορά την υπολογιστική αποδοτικότητα αυτών των γεννητριών μπορεί να βελτιωθεί με την επιλογή ορισμένων συντελεστών a_j ίσους με 0, 1 ή -1. Οι Deng και Lin το 2000 πρότειναν την Γρήγορη Γεννήτρια Πολλαπλής Αναδρομής όπου όλοι οι συντελεστές a_j ισούνται με το 0, εκτός του $a_1 = 1$ και του a_k , για το οποίο έδωσαν κατάλληλες τιμές για $k = 2, 3, 4, \dots$ και $m = 2^{31} - 1$.

Συμπερασματικά λόγω της μεγαλύτερης περιοδικότητας που τις χαρακτηρίζει οι MRGs αποτελούν καλές περιπτώσεις για την αντικατάσταση των γραμμικών γεννητριών ισοδυναμίας που χρησιμοποιούνται ακόμα σε διάφορες εφαρμογές.

2.5 Γεννήτρια Fibonacci Με Καθυστέρηση (Lagged Fibonacci Generators – LFG)

Μία ακολουθία Fibonacci περιγράφεται από την αναδρομική Σχέση

$$x_n = x_{n-1} + x_{n-2} \quad (15)$$

Για παράδειγμα αν αρχίσουμε με τις τιμές 0 και 1, καταλήγουμε στην ακολουθία 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ... Χρησιμοποιώντας το $\text{mod } m$ στους αριθμούς που παράγονται από αυτή σχηματίζεται η εξής αναδρομή

$$x_n = (x_{n-1} + x_{n-2}) \text{ mod } m \quad (16)$$

Οι αριθμοί που παράγονται από τη παραπάνω Σχέση παρόλο που είναι φαινομενικά τυχαίοι, στη πραγματικότητα δεν ικανοποιούν τις απαιτήσεις για την τυχειότητα λόγω της μεγάλης σειριακής συσχέτισης. Για να αντιμετωπιστεί αυτό το πρόβλημα συνδυάζονται αριθμοί όχι διαδοχικοί αλλά μεγαλύτερης απόστασης ώστε να παράγονται τυχαίοι αριθμοί. Οι γεννήτριες αυτές ονομάζονται Γεννήτρια Fibonacci Με Καθυστέρηση (Lagged Fibonacci Generators) και χαρακτηρίζονται από την γενική αναδρομή

$$x_n = (x_{n-j} + x_{n-k}) \bmod m \quad (17)$$

και ισχύουν οι περιορισμοί $0 \leq x_n \leq m-1$ και $0 < j < k < n$.

Αν το m είναι πρώτος αριθμός και ισχύει $k > j$, τότε η περίοδος της γεννήτριας επεκτείνεται στο $m^k - 1$. Μία σημαντική ιδιότητα συναντάμε στην περίπτωση που ο διαιρέτης m ισούται με μία δύναμη του 2 2^p . Η μέγιστη τότε περίοδος που μπορεί να επιτευχθεί τότε είναι $(2^k - 1)2^{p-1}$ αν από τους αριθμούς που θα χρησιμοποιηθούν για την αρχικοποίηση ένας τουλάχιστον είναι περιττός. Ένα σημαντικό πλεονέκτημα των LFGs όπως παρατηρεί κανείς είναι η πολύ μεγάλη περίοδος τους καθώς δεν εξαρτώνται αποκλειστικά από το m .

Υπάρχουν επιπλέον άλλες τρεις παραλλαγές της Σχέσης (17) όπου ο τελεστής $(+)$ αντικαθίσταται με έναν από τους τελεστές της αφαίρεσης $(-)$, του πολλαπλασιασμού $(*)$ και του αποκλειστικού ή (XOR) χωρίς το $\bmod m$. Στις περισσότερες περιπτώσεις αποφεύγεται ο τελεστής XOR , καθώς έχει αποδειχθεί ότι η γεννήτρια δεν εμφανίζει αποδοτική συμπεριφορά.

Η αποδοτικότητα αυτής της κατηγορίας γεννητριών εξαρτάται από τη επιλογή των παραμέτρων m , j και k . Ένα παράδειγμα αποδεδειγμένης καλής επιλογής παραμέτρων αποτελεί η γεννήτρια Extended Fibonacci Generator, η οποία προτάθηκε από τον Marsaglia το 1983. Οι τιμές των παραμέτρων είναι $k = 17$, $j = 5$ και η περίοδος της είναι ίση με $(2^{17} - 1)2^m$. Μερικοί άλλοι γνωστοί συνδυασμοί $\{j, k\}$ είναι $\{7, 10\}$, $\{24, 55\}$, $\{65, 71\}$, $\{128, 159\}$, $\{168, 521\}$, $\{418, 1279\}$.

Όσον αφορά την αρχικοποίηση οι LFGs παρουσιάζουν μία ευαίσθητη συμπεριφορά. Έχει αποδειχθεί ότι η αποδοτικότητα είναι μεγαλύτερη όταν χρησιμοποιείται μία αρχική σειρά $x_1, x_2, x_3, \dots, x_n$ καλά προσεγγμένων τιμών. Συμπερασματικά μία LFG παρουσιάζει επιτυχία την σημερινή εποχή αν και μετά από διερεύνηση διαπιστώθηκε ότι υπάρχουν κάποιες περιπτώσεις χαμηλής ποιότητας.

2.6 Γεννήτριες Ισοδυναμιών Με Πίνακες (Matrix Congruential Generators - MCG)

Οι Franklin, Grothe και Niederreiter πρότειναν μία ισοδυναμία που περιέχει πολλαπλασιασμό πινάκων ως βάση μίας γεννήτριας παραγωγής τυχαίων αριθμών. Αυτές οι γεννήτριες ονομάστηκαν ονομάστηκαν Γεννήτριες Ισοδυναμιών με Πίνακες (MCG) και είναι της μορφής

$$x_n = (Ax_{n-1} + c) \bmod m \quad (18)$$

όπου τα x_n , x_{n-1} , c είναι διανύσματα μήκους k , ενώ το A αποτελεί έναν πίνακα διαστάσεων $k \times k$.

Οι τιμές που περιέχουν τα διανύσματα και ο πίνακας είναι ακέραιοι με πεδίο τιμών ανάμεσα στο 1 και $m - 1$. Όσον αφορά το c , συνήθως είναι ένα διάνυσμα με όλα τα στοιχεία του ίσα με το 0 όπως συμβαίνει και σε μία MRG(Πολλαπλή Αναδρομική Γεννήτρια). Μία MCG παράγει μία ακολουθία διανυσμάτων με μέγιστη περίοδο $m^k - 1$ αν το m είναι πρώτος αριθμός.

Για να βελτιωθεί η αποδοτικότητα της Σχέσης (18) οι Deng και Lin σχεδίασαν έναν ειδικό πίνακα και ονόμασαν την γεννήτρια Γρήγορη Γεννήτρια Ισοδυναμιών Με Πίνακες (Fast Matrix Congruential Generator – FMCG). Ο πίνακας αυτός έχει την εξής δομή :

$$\begin{bmatrix} \alpha^1 & -1 & \dots & 0 \\ 0 & \alpha^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 \\ -1 & 0 & \dots & \alpha^k \end{bmatrix}$$

Μία πιο αναλυτική μορφή της Σχέσης (18) βάση των παραπάνω πίνακα είναι

$$\begin{bmatrix} X_{n,1} \\ X_{n,2} \\ \vdots \\ X_{n,k-1} \\ X_{n,k} \end{bmatrix} = \begin{bmatrix} \alpha^1 & -1 & \dots & 0 \\ 0 & \alpha^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 \\ -1 & 0 & \dots & \alpha^k \end{bmatrix} \begin{bmatrix} X_{n-1,1} \\ X_{n-1,2} \\ \vdots \\ X_{n-1,k-1} \\ X_{n-1,k} \end{bmatrix} \text{ mod } m$$

Τέλος μία επέκταση της MCG αποτελεί η :

$$x_n = (A_1 x_{n-1} + \dots + A_k x_{i-k}) \bmod m \quad (19)$$

Μία γεννήτρια που βασίζεται στη Σχέση (19) ονομάζεται Πολλαπλή Αναδρομική Τυχαία Γεννήτρια με Πίνακες (Multiple Recursive Matrix Random Generator). Με σωστή επιλογή των επιμέρους Πινάκων A_j ανάμεσα σε μηδενικούς, μοναδιαίους (ή αρνητικά μοναδιαίους) ή σε άλλες μορφές έχει δείξει ότι συνεισφέρει στην βελτίωση της υπολογιστικής αποδοτικότητας.

2.7 Μη Γραμμικές Γεννήτριες

Οι γραμμικές γεννήτριες παρόλο που είναι γρήγορες, με απλές πράξεις και υπολογισμούς παρουσιάζουν παρουσιάζουν κάποια ανεπιθύμητα χαρακτηριστικά λόγω της γραμμικότητας της αναδρομής τους. Για αυτό τον λόγο προτάθηκαν και αναλύθηκαν αρκετές μη γραμμικές γεννήτριες ισοδυναμιών. Η πιο απλή τέτοια γεννήτρια ορίζεται από τον αναδρομικό τύπο

$$x_n = \alpha x_{n-1} (x_{n-1} + c) \bmod m, \quad 0 \leq x_n < m \quad (20)$$

Παρακάτω θα παρουσιαστούν οι κυριότερες γεννήτριες αυτής της κατηγορίας.

2.7.1 Αντίστροφες Γεννήτριες Ισοδυναμιών (Inversive Congruential Generator – ICG)

Οι αντίστροφες γεννήτριες ισοδυναμιών παρουσιάστηκαν από τους Eichenauer και Lehn το 1986 και χρησιμοποιούν την ιδιότητα τους πολλαπλασιαστικού αντιστρόφου modulo (modulo multiplicative inverse), όταν αυτό είναι εφικτό, για να παράγουν τις τιμές κάθε ακολουθίας. Η γενική αναδρομική μορφή των ICGs είναι η :

$$x_n = (ax_{n-1}^{-1} + c) \bmod m \quad (21)$$

όπου το x^{-1} συμβολίζει τον πολλαπλασιαστικό αντίστροφο του $x \bmod m$. Εάν υπάρχει υπολογίζεται από την σχέση

$$1 = x^{-1}x \bmod m \quad (22)$$

για όλες τις μη μηδενικές τιμές του x , οι οποίες είναι σχετικά πρώτες με το m . Αν ο πολλαπλασιαστικός αντίστροφος δεν υπάρχει το x^{-1} παίρνει την τιμή 0. Η μέγιστη περίοδος των ICGs ισούται με το m .

Το 1994, οι Eichenauer – Hermann και Ickstadt πρότειναν μια μη γραμμική γεννήτρια, η οποία παράγει περιττούς ακραίους στο διάστημα $[0, m - 1]$. Αργότερα παρουσίασαν μία παραλλαγή της όπου εμφανίζονται στην έξοδο όλοι οι μη αρνητικοί ακέραιοι έως το $m - 1$. Ονόμασαν αυτή την παραλλαγή explicit inversive congruential και ο τύπος της είναι ο εξής

$$x_n = n(an + b)^{-1} \bmod m \quad (23)$$

όπου $n = 0, 1, 2, \dots$, $a = 2 \bmod 4$, $b = 1 \bmod 2$. Η περίοδος της συγκεκριμένης γεννήτριας ισούται με το m το οποίο αποτελεί δύναμη του δύο, 2^k για $k \geq 4$.

Οι πολύπλοκες υπολογισμοί που τις χαρακτηρίζουν τις καθιέρωσαν ως βασικές μεθόδους για την παραγωγή τυχαίων αριθμών. Μάλιστα σε ορισμένες περιπτώσεις ξεπερνούν σε τυχειότητα τις γραμμικές. Ωστόσο σε ελέγχους που πραγματοποιήθηκαν, αδυναμίες που περιορίζουν της χρήση τους.

2.7.2 Επιπλέον Μη Γραμμικές Γεννήτριες Ισοδυναμιών

Οι γεννήτριες που είδαμε στην προηγούμενη παράγραφο δεν είναι οι μοναδικές κατηγορίες μη γραμμικών ισοδυναμιών. Ο Knuth παρουσίασε μία μη γραμμική γεννήτρια που ορίζεται από την γενική αναδρομική σχέση

$$x_n = (dx_{n-1}^2 + ax_{n-1} + c) \bmod m, \quad 0 \leq x_n \leq m \quad (24)$$

Ενώ μία από τις πιο διαδεδομένες μεθόδους στο κλάδο της κρυπτογραφίας λόγω της κρυπτογραφικής της ισχύς είναι η γεννήτρια Blum Blum Shub (BBS) η οποία πήρε το όνομα της από τους εμπνευστές της Lenore Blum, Manuel Blum και Michael Shub το 1986.

Η γενική μορφή που την χαρακτηρίζει είναι η εξής

$$x_n = x_{n-1}^2 \bmod m \quad (25)$$

Η διαδικασία ξεκινάει με την επιλογή δύο μεγάλων πρώτων αριθμών p και q , οι οποίοι θα πρέπει να διαιρούνται με το 4 και να δίνουν υπόλοιπο 3

$$p \bmod 4 = q \bmod 4 = 3$$

Το γινόμενο των p και q αποτελεί τον διαιρέτη m της σχέσης.

Στην συνέχεια επιλέγεται ένας τυχαίος αριθμός s τέτοιος ώστε οι τιμές των s και m να είναι αριθμοί πρώτοι μεταξύ τους. Επιπλέον η αρχική τιμή x_0 υπολογίζεται ως

$$x_0 = s^2 \bmod m \quad (26)$$

Τέλος στην έξοδο εμφανίζεται μία ακολουθία από bits $b_1b_2b_3\dots$ όπου $b_n = 0$ εάν ο συντελεστής x_n της σχέσης (26) είναι άρτιος και $b_n = 1$ αν το x_n είναι περιττό. Επομένως σε κάθε επανάληψη λαμβάνουμε τελικά το λιγότερο σημαντικό bit για κάθε x_n .

Ο παρακάτω πίνακας απεικονίζει ένα παράδειγμα της BBS όπου ισχύει $m = 383 * 503 = 192649$ και $s = 101355$

n	x_n	b_n
0	20749	
1	143135	1
2	177671	1
3	97048	0
4	89992	0
5	174051	1

6	80649	1
7	45663	1
8	69442	0
9	186894	0
10	177046	0

Πίνακας 2.3: Αποτελέσματα δέκα επαναλήψεων μίας BBS για $m=192649$ και $s = 101355$

Όπως παρατηρεί κανείς η έξοδος της γεννήτριας αυτής δεν μπορεί να υπολογιστεί σε πολυωνυμικό χρόνο, αν δεν είναι γνωστά τα p και q . Αυτή η δυσκολία παραγοντοποίησης του m αποτελεί την ασφάλεια της BBS.

2.8 Linear Feedback Shift Register Γεννήτριες

Οι Feedback Shift register γεννήτριες (FSR) παρουσιάστηκαν από τον Tausworthe το 1965 και καλούνται γραμμικές (LFSR) όταν έχουν τη γενική μορφή

$$x_n = (a_1 x_{n-1} + \dots + a_k x_{n-k}) \bmod 2 \quad (27)$$

για κάποιους προκαθορισμένους συντελεστές $a_1, a_2, \dots, a_{k-1}$ που παίρνουν τιμές $\{0, 1\}$ και επιπλέον ισχύει $a_k = 1$. Τα x_n που παράγονται ανήκουν επίσης στο δυαδικό αλφάβητο (F_2) για μία ακολουθία από 0 και 1 μήκους k ως αρχικό σπόρο.

Όσον αφορά την μέγιστη περίοδο που παρουσιάζει, αυτή ισούται με $2^k - 1$, αν και μόνο αν το χαρακτηριστικό πολυώνυμο που καθορίζει την Σχέση (27)

$$P(z) = - \sum_{i=0}^k \alpha_i z^{k-i} \quad (28)$$

είναι πρωτεύον πολυώνυμο όταν $\alpha_0 = -1$ και το διάνυσμα του σπόρου δεν έχει όλες τις τιμές ίσες με το 0. Στην ειδική περίπτωση που το αρχικό διάνυσμα αποτελείται μόνο από 0 ο κύκλος της γεννήτριας ισούται με 1. Σε κάθε άλλη περίπτωση θα είναι μεγαλύτερος της μονάδας.

Για να μειώσουμε την υπολογιστική πολυπλοκότητα και να αυξήσουμε την ταχύτητα, οι περισσότεροι συντελεστές α_j για $0 < j < k$ παίρνουν την τιμή 0. Για αυτό τον λόγο συνήθως η Σχέση (27) απλοποιείται σε

$$x_n = (x_{n-k} + x_{n-k+q}) \bmod 2 \quad (29)$$

Η παραπάνω ισοδυναμία οδηγεί σε τριώνυμα της μορφής $x^k + x^r + 1$ όπου $r = k - q$ ακολουθούμενα από το modulo 2. Υπάρχουν και άλλες ισοδυναμίες που οδηγούν σε ανώτερης τάξης πολυώνυμα που δεν θα αναλυθούν εδώ. Μία ακόμα απλοποίηση της (29) είναι η αντικατάσταση της πρόσθεσης ακολουθούμενη από το modulo 2 με την πράξη αποκλειστικό ή (XOR) που συμβολίζεται με $\oplus$ και είναι ισοδύναμη. Επομένως η Σχέση (29) μετατρέπεται ως

$$x_n = (x_{n-k} \oplus x_{n-k+q}) \text{ , όπου } 0 < q < k < n \quad (30)$$

Για παράδειγμα η ισοδυναμία (τ) μίας LFSR για τις τιμές $k = 4$ και $q = 3$ γίνεται

$$x_n = (x_{n-4} \oplus x_{n-1})$$

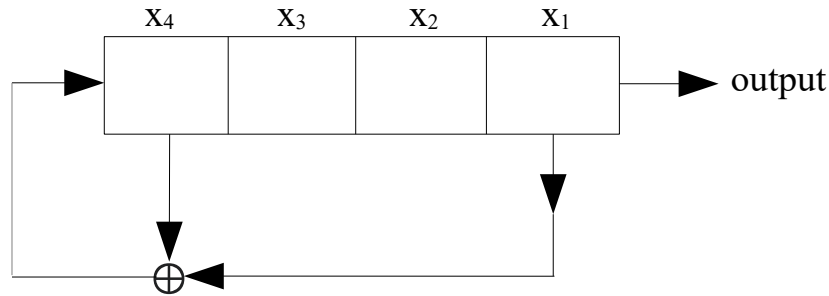
και το χαρακτηριστικό της πολυώνυμο είναι $x^4 + x^1 + 1$.

Ακολουθεί ένας πίνακας με τα αποτελέσματα 15 επαναλήψεων για σπόρο ίσο με $[1, 0, 1, 0]$

n	$(x_{n-4} \oplus x_{n-1})$	x_n
1	—	—
2	—	—
3	—	—
4	—	—
5	$(x_1 \oplus x_4)$	1
6	$(x_2 \oplus x_5)$	1
7	$(x_3 \oplus x_6)$	0
8	$(x_4 \oplus x_7)$	0
9	$(x_5 \oplus x_8)$	1
10	$(x_6 \oplus x_9)$	0
11	$(x_7 \oplus x_{10})$	0
12	$(x_8 \oplus x_{11})$	0
13	$(x_9 \oplus x_{12})$	1
14	$(x_{10} \oplus x_{13})$	1
15	$(x_{11} \oplus x_{14})$	1

Πίνακας 2.4: Αποτελέσματα δεκαπέντε επαναλήψεων μίας LFSR για $k=4$, $r = 1$ και σπόρο $[1,0,1,0]$

Τέλος μία LFSR μπορεί να υλοποιηθεί με την βοήθεια ενός ανατροφοδοτικού καταχωρητή ολίσθησης (Feedback Shift Register). Στο επόμενο σχήμα φαίνεται η υλοποίηση του παραπάνω παραδείγματος με την βοήθεια ενός καταχωρητή ολίσθησης και της δυαδικής πράξης XOR ($\oplus$).



Σχήμα 2.2: Αναπαράσταση μίας LFSR με χρήση καταχωρητή ολίσθησης

2.9 Generalized Feedback Shift Register Γεννήτριες

Μία παραλλαγή της γεννήτριας του Tausworthe πρότειναν οι Lewis και Payne, την οποία ονόμασαν Generalized Feedback Shift Register (GFSR). Η βασική ιδέα είναι ίδια με αυτή των LFSRs μόνο που αντί να χρησιμοποιηθούν κάποια bits για την δημιουργία του επόμενου στην ακολουθία, δημιουργείται μία ακολουθία από διανύσματα λέξεων χρησιμοποιώντας προηγούμενες λέξεις κατασκευασμένες με κάποια καθυστέρηση.

Η αναδρομή που βασίζεται μία GFSR είναι η

$$W_n = (W_{n-k} \oplus W_{n-k+q}) \quad (31)$$

όπου W_n ένα διάνυσμα λέξης μήκους ℓ .

Έστω μία ακολουθία x_i από bits που έχει παραχθεί από μία LFSR για ένα χαρακτηριστικό πολυώνυμο $x^k + x^r + 1$. Χρησιμοποιώντας αυτή την ακολουθία κατασκευάζονται τα διανύσματα λέξεων $W_1, W_2, \dots, W_k$ των οποίων τα bits επιλέγονται από την ακολουθία x_i με κάποια καθυστέρηση d θέσεων bit. Κάθε bit της λέξης επιλέγεται με καθυστέρηση d του προηγούμενου της. Στη συνέχεια με βάση την Σχέση (31) δημιουργούνται οι επόμενες λέξεις.

Για να γίνει πιο κατανοητή η διαδικασία, ακολουθεί ένα παράδειγμα με ακολουθία $x_i = 101011001000111$ για το προηγούμενο χαρακτηριστικό πολυώνυμο $x^4 + x^1 + 1$. Για καθυστέρηση ίση με 3 δημιουργούνται οι πρώτες 4 λέξεις μήκους 4-bits.

$$W_1 = 1000, W_2 = 0100, W_3 = 1110, W_4 = 0001$$

Οι επόμενες λέξεις W_n θα βρεθούν με χρήση της Σχέσης (31)

$$W_5 = (W_1 \oplus W_4) = 1001$$

$$W_6 = (W_2 \oplus W_5) = 1101$$

$$W_7 = (W_3 \oplus W_6) = 0011$$

...

Οι Matsumoto και Kurita πρότειναν μία παραλλαγή των GFSR όπου τα διανύσματα από bits μήκους ℓ που παράγονται από τον όρο W_{n-k+q} της σχέσης (31) μετατοπίζονται πριν εκτελεστεί η δυαδική πράξη XOR. Για να υλοποιηθεί αυτό κάθε ένα από αυτά τα διανύσματα πολλαπλασιάζονται με ένα πίνακα A $\ell \times \ell$ διαστάσεων.

$$W_n = (W_{n-k} \oplus W_{n-k+q} A) \quad (32)$$

Οι εισηγητές της πρότειναν μάλιστα τρόπους για να επιλέγονται κατάλληλοι πίνακες A ώστε η διαδικασία να γίνεται πιο απλή και γρήγορη. Η συγκεκριμένη γεννήτρια έχει γίνει γνωστή ως Twisted GFSR.

2.10 Συνδυαστικές Γεννήτριες

Για να επιτευχθεί αύξηση στη περίοδο και να υπάρχει βελτίωση στην αποδοτικότητα των ψευδοτυχαίων γεννητριών χρησιμοποιείται συχνά συνδυασμός διαφορετικών γεννητριών. Συνήθως επιλέγονται ένας ή παραπάνω αλγόριθμοι για να εκτελέσουν κάποιο συμπληρωματικό ρόλο για μία κύρια γεννήτρια. Η πρώτη συνδυαστική προτάθηκε από τους MacLaren και Marsaglia το 1965, οι οποίοι χρησιμοποίησαν περισσότερες από μία γεννήτριες ώστε να δημιουργήσουν έναν πίνακα με τυχαίες τιμές.

Ένα σημαντικό χαρακτηριστικό που τις διακρίνει είναι ότι υπάρχει η δυνατότητα να συνδυαστούν γεννήτριες διαφορετικών τύπων. Στην πρακτική γενικά άλλωστε χρησιμοποιούνται δύο γεννήτριες, όπου η μία παράγει κάποια ακολουθία από τυχαίους αριθμούς και η δεύτερη επιλέγει ποιοι αριθμοί αυτής της ακολουθίας θα εμφανιστούν στην έξοδο. Με αυτό τον τρόπο διατηρούνται τα χαρακτηριστικά της κάθε ανεξάρτητης γεννήτριας και επιπλέον αυξάνεται η περίοδος. Παρακάτω φαίνονται μερικές γνωστές συνδυαστικές γεννήτριες

- Wichmann – Hill : συνδυάζει 3 διαφορετικές LCGs, δεχόμενη σαν σπόρους διανύσματα μήκους 3. Επιτυγχάνει περίοδο της τάξης 10^{12} .
- Ranmar : χρησιμοποιεί μία LFG μαζί με μία σειρά απο πρώτους αριθμούς. Πετυχαίνει περίοδο ίση με 2^{144} και παρέχει πολύ γρήγορη υλοποίηση
- Super – Duper : αποτελείται από μία LCG σε συνδυασμό με μία Tausworthe γεννήτρια. Παρουσιάζει περίοδο ίση με $4.6 \cdot 10^{18}$ για τους περισσότερους σπόρους. Δέχεται σαν σπόρο δυο διαφορετικούς ακεραίους, με τον δεύτερο να είναι απαραίτητα περιττός.
- Mersenne Twister : Αναπτύχθηκε από τους Matsumoto και Nishimura και αποτελεί μία twisted GFSR. Περισσότερες πληροφορίες στον επόμενο Κεφάλαιο.

Κεφάλαιο 3^ο

MERSENNE TWISTER

3.1 Η Γεννήτρια Mersenne Twister

Η γεννήτρια Mersenne Twister ανήκει στην κατηγορία των γεννητριών ψευδοτυχαίων αριθμών (PRNGs) και αποτελεί την πιο ευρέως διαδιδόμενη γενικού σκοπού που υπάρχει. Αναπτύχθηκε το 1997 από τους Makoto Matsumoto και Takuji Nishimura και πήρε το όνομα της από την χρήση των Mersenne

primes . Σχεδιάστηκε ώστε να αποφύγει τα πολλά μειονεκτήματα που παρουσίαζαν οι γεννήτριες τυχαίων αριθμών μέχρι τότε , μάλιστα είναι η πρώτη που μπόρεσε να παράγει τόσο υψηλής ποιότητας αριθμούς σε τόσο μικρό χρονικό διάστημα. Βασίζεται στις γεννήτριες γραμμικών ισοδυναμιών και η πιο σημαντική υλοποίηση της φέρει το όνομα MT19937 από τον πρώτο $2^{19937}-1$ χρησιμοποιώντας 32 – bit μήκος λέξης . Υπάρχει και άλλη μία υλοποίηση που χρησιμοποιεί 64 – bit μήκος λέξης , MT19937 – 64 η οποία παράγει διαφορετική ακολουθία τυχαίων αριθμών. Η MT19937 αποτελεί την προκαθορισμένη ψευδοτυχαία γεννήτρια σε πολλά λογισμικά συστήματα όπως Microsoft Excel, Maple , C compilers , Ruby , Python , Matlab , php και άλλα .

3.2 Αλγοριθμική Ανάλυση

Ο αλγόριθμος Mersenne Twister είναι μία twisted GFSR και βασίζεται στην γραμμική αναδρομική συνάρτηση

$$X_{k+n} = X_{k+m} \oplus ((X_k^u \parallel X_{k+1}^l)A) \quad k = 0, 1, 2, \dots \quad (33)$$

και παράγει ακολουθίες από διανύσματα μήκους w τα οποία περιέχουν τιμές που ανήκουν στο σύνολο $F_2 = \{0, 1\}$. Ένα τέτοιο διάνυσμα στην επιστήμη των υπολογιστών ονομάζεται διάνυσμα λέξης (word) μήκους w . Οι λέξεις αυτές σχηματίζουν αριθμούς στο δεκαδικό σύστημα μεταξύ το 0 και $2^w - 1$. Διαιρώντας αυτούς τους αριθμούς με το $2^w - 1$ μετατρέπονται σε πραγματικούς ανάμεσα στο 0 και 1.

Η παραπάνω αναδρομική συνάρτηση περιέχει αρκετές σταθερές, έναν ακέραιο n ο οποίος αποτελεί τον βαθμό της αναδρομής, έναν ακέραιο

r που χρειάζεται για τον υπολογισμό του x_k^u και x_{k+1}' με $0 \leq r \leq w-1$, έναν m όπου $1 \leq m \leq n$ και ένα σταθερό πίνακα A διαστάσεων $w \times w$ με τιμές στο $F_2 = \{0, 1\}$.

Ο αλγόριθμος Mersenne Twister τροφοδοτείται με την ακολουθία $x_0, x_1, x_2, \dots, x_{n-1}$ ως αρχικούς σπόρους και παράγει x_n με τη χρήση της Σχέσης () για $k = 0$. Για $k = 1, 2, \dots, n$ η γεννήτρια παράγει τις επόμενες $x_{n+1}, x_{n+2}, \dots$.

Στη δεξιά πλευρά της αναδρομής (33) το x_k^u αποτελεί τα πρώτα $w-r$ bits (upper $w-r$ bits) του x_k ενώ το x_{k+1}' ισούται με τα τελευταία r bits (lower r bits) του x_{k+1} . Έτσι αν $x = [x_{w-1}, x_{w-2}, \dots, x_0]$ το x^u είναι τα $[x_{w-1}, x_{w-2}, \dots, x_r]$ bits και το αποτελείται από τα $[x_{r-1}, x_{r-2}, \dots, x_0]$. Το $(x_k^u \parallel x_{k+1}')$ δεν είναι τίποτα άλλο από μία λέξη που δημιουργήθηκε από την ένωση του x_k^u με το x_{k+1}' , με το $\parallel$ να συμβολίζει αυτή την ένωση (concatenation). Στη συνέχεια πολλαπλασιάζεται αυτό το διάνυσμα της ένωσης με τον πίνακα A για να δημιουργηθεί ένα νέο στο οποίο εφαρμόζεται η δυαδική πράξη αποκλειστικού ή (XOR – πρόσθεση modulo 2) με το x_{k+m} .

Όσον αφορά τον πίνακα A επιλέγεται μία μορφή αυτού ώστε ο πολλαπλασιασμός xA να είναι πιο εύκολος και πιο γρήγορος. Μία υποψήφια μορφή αποτελεί η παρακάτω

$$A = \begin{bmatrix} & & 1 & & & \\ & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \\ \alpha_{w-1} & \alpha_{w-1} & \dots & \dots & \alpha_0 & \end{bmatrix}$$

Με την επιλογή αυτού του πίνακα ο υπολογισμός xA μπορεί να πραγματοποιηθεί μόνο με τις δυαδικές πράξεις της δεξιάς ολίσθησης ($\gg$) και του αποκλειστικού ή ($\oplus$). Αν $x_0 = 0$ τότε εκτελείται δεξιά ολίσθηση στο διάνυσμα $x = [x_{w-1}, x_{w-2}, \dots, x_0]$ ενώ αν $x_0 = 1$ εκτελείται δεξιά ολίσθηση στο διάνυσμα x και στη συνέχεια αποκλειστικό ή με την τελευταία γραμμή του πίνακα A .

Συνοψίζοντας ισχύει

$$\begin{aligned} xA &= \text{shiftright}(x) \quad \text{αν } x_0 = 0 \\ \text{ή} \\ xA &= \text{shiftright}(x) \oplus \alpha \quad \text{αν } x_0 = 1 \end{aligned}$$

όπου $\alpha = [\alpha_{w-1}, \alpha_{w-2}, \dots, \alpha_0]$ η τελευταία γραμμή του πίνακα A .

Τέλος το διάνυσμα x που έχει παραχθεί από την αναδρομική Σχέση (33) πολλαπλασιάζεται με ένα κατάλληλο $w \times w$ πίνακα T που ονομάζεται *tempering* από τους Matsumoto και Kurita. Για τον πολλαπλασιασμό αυτό xT όπως και για εκείνο με τον πίνακα A επιλέγεται μία σειρά απο δυαδικές πράξεις αποκλειστικού ή ($\oplus$), δυαδική πρόσθεση (AND) και ολισθήσεις δεξιά ($\gg$) και αριστερά ($\ll$).

$$\begin{aligned} y &= x \oplus (x \gg u) \\ y &= y \oplus ((y \ll s) \text{ AND } b) \\ y &= y \oplus ((y \ll t) \text{ AND } c) \\ z &= y \oplus (y \gg \ell) \end{aligned}$$

όπου ℓ, s, t και u ακέραιοι, b και c κατάλληλα bitmasks μεγέθους w .

Εδώ να σημειώσουμε ότι σε αντίθεση με άλλες γεννήτριες στο F_2 που βασίζονται σε τριώνυμα και παρουσιάζουν χαμηλή τυχαιότητα, το χαρακτηριστικό πολυώνυμο του αλγορίθμου Mersenne Twister αποτελείται από πολλούς παράγοντες και αυτό αυξάνει την τυχαιότητα των ακολουθιών που παράγονται σε μεγάλο βαθμό.

3.3 Παρουσίαση του Mersenne Twister σε Φυσική Γλώσσα

Σε αυτή την παράγραφο θα παρουσιαστεί η διαδικασία που ακολουθεί ο αλγόριθμος Mersenne Twister σε μία φυσική γλώσσα κατά βήματα ώστε να γίνει πιο κατανοητή η λειτουργία του.

Έστω up το bitmask των πρώτων $w - r$ bits, ℓ_0 το bitmask των τελευταίων r bits και a η τελευταία γραμμή του πίνακα A .

Βήμα 1 : $k \leftarrow 0$

// εκχώρηση του 0 σε μία μεταβλητή k

$x[0], x[1], \dots, x[n-1]$

// αρχικές μη μηδενικές τιμές

Βήμα 2 : $y \leftarrow (x[k] \text{ AND } up) \text{ OR } (x[(k+1) \bmod n] \text{ AND } \ell_0)$

// υπολογισμός του όρου $(x_k^u \parallel x_{k+1}^{\ell})$ και εκχώρηση στο y

Βήμα 3: // Υπολογισμός του πολλαπλασιασμού xA

$x[k] \leftarrow x[(k+m) \bmod n] \text{ XOR } (y \gg 1)$

// αν το λιγότερο σημαντικό bit του y ισούται με 0

ή
 $x[k] \leftarrow x[(k + m) \bmod n] \text{ XOR } ((y \gg 1) \text{ XOR } \alpha)$
 // αν το λιγότερο σημαντικό bit του y ισούται με 1

Βήμα 4: // υπολογισμός $x[k]T$
 $y \leftarrow x[k]$
 $y \leftarrow y \text{ XOR } (y \gg s)$
 $y \leftarrow y \text{ XOR } ((y \ll s) \text{ AND } b)$
 $y \leftarrow y \text{ XOR } ((y \ll t) \text{ AND } c)$
 $y \leftarrow y \text{ XOR } (y \ll \ell)$
 έξοδος y

Βήμα 5: $k \leftarrow (k + 1) \bmod n$

Βήμα 6: Επιστροφή στο βήμα 2

3.4 Παράμετροι και Περιοδικότητα

Όπως και σε άλλες ψευδοτυχαίες γεννήτριες που είδαμε στο προηγούμενο κεφάλαιο η σωστή επιλογή παραμέτρων είναι απαραίτητη ώστε να επιτευχθεί η μέγιστη περίοδος $2^{nw-r} - 1$. Παρακάτω αναφέρονται οι παράμετροι που χρησιμοποιούνται στη γεννήτρια MT19937.

Παράμετρος	Τιμή
w	32
n	624

m	397
r	31
a	9908B0DF ₁₆
u	11
d	FFFFFFFF ₁₆
s	7
b	9D2C5680 ₁₆
t	15
c	EFC60000 ₁₆
ℓ	18

Πίνακας 3.1: Παράμετροι της γεννήτριας MT19937

Με αυτές τις παραμέτρους η συγκεκριμένη γεννήτρια επιτυγχάνει την τεράστια περίοδο $2^{19937} - 1$ χρησιμοποιώντας 624 διανύσματα λέξης (words) μήκους $w = 32$ με πεδίο $[0, 2^{32} - 1]$ σε κάθε επανάληψη. Αυτή η τιμή είναι μεγαλύτερη με τεράστια διαφορά από άλλες γεννήτριες. Καθώς επίσης σε συνδυασμό με την μεγάλη τυχαιότητα που παρουσιάζουν οι παραγόμενες ακολουθίες καθιστούν την MT19937 μία από τις πιο διαδεδομένες γεννήτριες παραγωγής ψευδοτυχαίων αριθμών την σημερινή εποχή.

3.5 Παρουσίαση του Mersenne Twister με Ψευδοκώδικα

Παρακάτω φαίνεται ο αλγόριθμος Mersenne Twister σε μορφή ψευδοκώδικα. Ακολουθώντας τα βήματα που είδαμε προηγουμένως δημιουργούνται τρεις συναρτήσεις `init_seed()` η οποία αρχικοποιεί τις $n -$

1 τιμές για κάποιο αριθμό `seed` που δόθηκε ως παράμετρος, την συνάρτηση `twist()` η οποία δημιουργεί τα επόμενα 624 διανύσματα λέξεων με βάση την Σχέση () και τέλος την συνάρτηση `extract_number()` η οποία πραγματοποιεί τον πολλαπλασιασμό με το tempering πίνακα και καλεί την `twist()` όταν είναι απαραίτητο.

```
int[0..n-1] MT
int index = n+1
const int lower_mask = (1 << r) - 1
const int upper_mask = lowest w bits of (NOT lower_mask)

function init_seed(int seed) {
    index = n
    MT[0] = seed
    for i from 1 to (n - 1) {
        MT[i] = lowest w bits of (f * (MT[i-1] XOR (MT[i-1] >> (w-2))) + i)
    }
}

function extract_number() {
    if index >= n {
        twist()
    }

    int y = MT[index]
    y = y XOR ((y >> u) AND d)
    y = y XOR ((y << s) AND b)
    y = y XOR ((y << t) AND c)
    y = y XOR (y >> l)

    index = index + 1
    return lowest w bits of (y)
}
```

```

function twist() {
    for i from 0 to (n-1) {
        int x = (MT[i] AND upper_mask)
                + (MT[(i+1) MOD n] AND lower_mask)
        int xA = x >> 1
        if (x MOD 2) != 0 {
            xA = xA XOR a
        }
        MT[i] = MT[(i + m) MOD n] XOR xA
    }
    index = 0
}

```

Κεφάλαιο 4^ο

ΕΦΑΡΜΟΓΗ SLOT MACHINE

4.1 Εισαγωγή

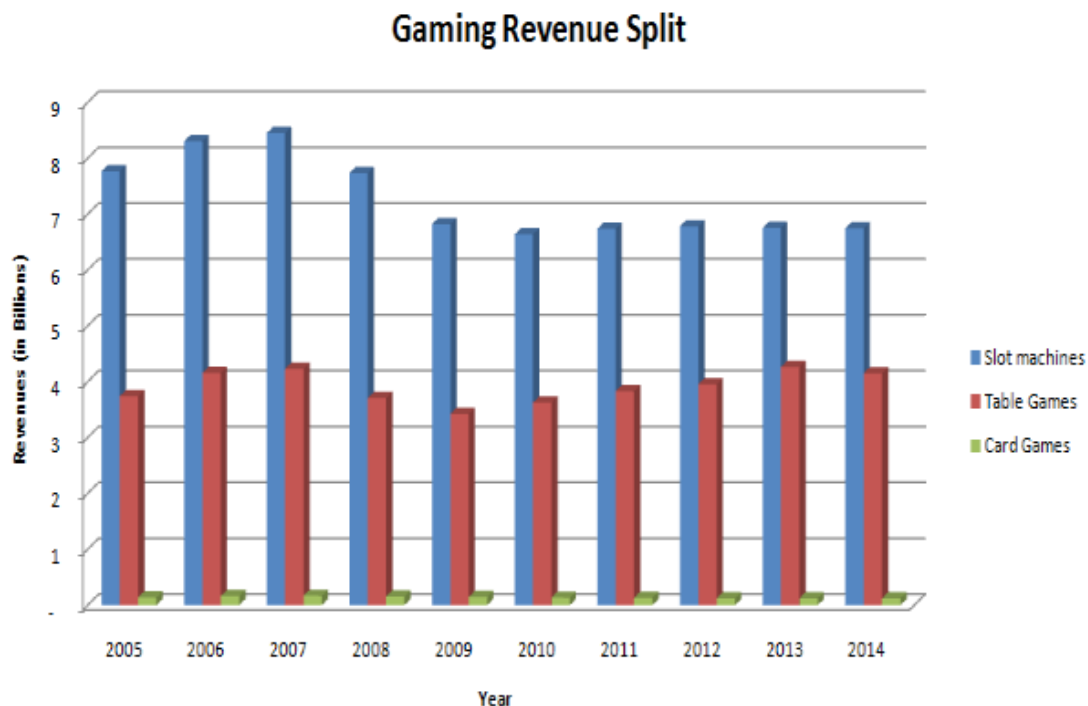
Στα πλαίσια της συγκεκριμένης εργασίας, εκτός από την ανάλυση των ψευδοτυχαίων γεννητριών παραγωγής τυχαίων αριθμών, θεωρείται αναγκαία η ανάπτυξη και υλοποίηση μίας ρεαλιστικής εφαρμογής που κάνει χρήση τις ιδιότητες αυτών. Προτιμήθηκε η επιλογή αυτής της εφαρμογής να γίνει από τον χώρο των τυχερών παιχνιδιών στα οποία βασίζονται τα καζίνο και γενικά η βιομηχανία τυχερών παιχνιδιών και στοιχηματισμού.



Η βιομηχανία στοιχηματισμού έχει γνωρίσει ραγδαία ανάπτυξη από την στιγμή που νομιμοποιήθηκε. Ειδικά τις τελευταίες δυο δεκαετίες λόγω της εξέλιξης του διαδικτυακών παιχνιδιών και τα ποσά που δαπανιούνται σε αυτά η αξίας της στην παγκόσμια αγορά έχει αυξηθεί σε μεγάλο βαθμό.

Σήμερα υπολογίζεται ότι φτάνει τα 443 δισ. Δολάρια το χρόνο και εκτιμάτε ότι θα φτάσει μέχρι το 2027 στα 647 δισ.

Το πιο δημοφιλές τυχερό παιχνίδι στα καζίνο και στο διαδίκτυο παγκοσμίως είναι οι μηχανές slot (slot machines) ή όπως είναι γνωστές στους περισσότερους " φρουτάκια ". Αποτελεί το πιο κερδοφόρο παιχνίδι ξεπερνώντας κατά πολύ τα υπόλοιπα όπως το blackjack, video poker, ρουλέτα και άλλα. Ειδικά για τα καζίνο αποτελούν το ποσοστό 50% ή και περισσότερο των συνολικών κερδών τους.



Η πρώτη μηχανή slot δημιουργήθηκε από τον γεννημένο στην Αμερική μηχανικό Charles August Fey το 1894 στο San Francisco. Στις μέρες μας οι παραδοσιακές ηλεκτρομηχανικές μηχανές slot δεν χρησιμοποιούνται ή και να χρησιμοποιούνται οι περιπτώσεις είναι ελάχιστες. Οι σύγχρονες μηχανές slot όπως και άλλες μηχανές που απαιτούν τυχαίους αριθμούς για την λειτουργία τους περιέχουν στην κατασκευή τους γεννήτριες παραγωγής ψευδοτυχαίων αριθμών.

Στην συνέχεια του κεφαλαίου θα παρουσιαστούν οι κανόνες, οι όροι, οι πιθανότητες και η υλοποίηση μιας ενδεικτικής μηχανής slot.

4.2 Κανόνες Και Χαρακτηριστικά Παιχνιδιού

Πριν δούμε πως μπορεί κάποιος να παίξει με μία μηχανή slot, θα πρέπει να γίνει κατανοητό ότι το αποτέλεσμα κάθε περιστροφής ενός τροχού είναι τυχαίο ή πιο σωστά ψευδότυχαio αφού προέρχεται από μία

PRNG και δεν εξαρτάται από άλλους παράγοντες. Κάθε γύρισμα των τροχών είναι ανεξάρτητο, το οποίο σημαίνει ότι δεν επηρεάζεται από τι συνέβη την προηγούμενη φορά που γύρισαν Αυτό ισχύει και για τα δύο είδη που υπάρχουν (online, φυσικές).

Γενικά οι μηχανές slot δεν είναι πολύπλοκες. Ως ένα παιχνίδι τύχης δεν υπάρχουν πολλά που πρέπει να ξέρει κάποιος για να παίζει. Η διαδικασία είναι απλή, εισάγει κάποιος το συνολικό ποσό που επιθυμεί, προσαρμόζει το ποσό που θέλει να στοιχηματίσει, πατάει το κουμπί εκκίνησης και οι τροχοί ξεκινούν να γυρίζουν. Όταν σταματήσουν αν υπάρχει κάποιος νικητήριος συνδυασμός στη οθόνη κερδίζει αλλιώς χάνει το ποσό που στοιχημάτισε. Ακόμα και αν ο παίχτης δεν μπορεί να καταλάβει τι συμβαίνει στην οθόνη η διαδικασία είναι τόσο απλή που μπορεί να παίζει.

Υπάρχουν κάποια χαρακτηριστικά που είναι απαραίτητα για τον σχεδιασμό μίας μηχανής slot. Αυτά είναι τα εξής :

1. Το πλήθος των τροχών (reels) : Συνήθως αποτελούνται από 3 ή 5 τροχούς.
2. Το συνολικό πλήθος συμβόλων σε κάθε τροχό : Συνήθως είναι 22 ή 32 σύμβολα σε κάθε τροχό.
3. Τα σύμβολα : Υπάρχουν 5 κατηγορίες από τις οποίες επιλέγονται τα σύμβολα που καλύπτουν τις θέσεις του κάθε τροχού. Κάποιες από αυτές τις κατηγορίες έχουν κάποιες ιδιότητες.

- Βασικά Σύμβολα

Οι περισσότερες θέσεις αν όχι όλες καλύπτονται με αυτά και αποτελούν την βάση των νικητήριων συνδυασμών. Δεν είναι απαραίτητο να είναι μοναδικά. Τις περισσότερες φορές ένα βασικό σύμβολο εμφανίζεται περισσότερες από μία φορές.

- "wild" Σύμβολα

Μπορούν να πάρουν την θέση ενός άλλου συμβόλου ώστε να συμπληρωθεί ένας συνδυασμός που κερδίζει με μόνη εξαίρεση ένα "jackpot" σύμβολο. Επίσης συνήθως καλύπτουν τις ιδιότητες των βασικών συμβόλων δηλαδή για 3 τροχούς αν ο συνδυασμός είναι μόνο "wild" σύμβολα θα είναι ένας νικητήριος συνδυασμός.

- " bonus " Σύμβολα

Αυτά μπορούν να εκτελούν διαφορετικές ενέργειες όταν εμφανίζονται σε ένα τροχό όπως "bonus" παιχνίδι ή δωρεάν γυρίσματα των τροχών.

- " jackpot " Σύμβολο

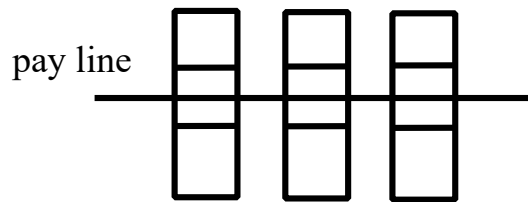
Ένας συνδυασμός που αποτελείται μόνο από αυτό το σύμβολο προσφέρει τον μεγαλύτερο κέρδος στον παίκτη. Συνήθως υπάρχει μόνο μία φορά σε κάθε τροχό αφού αποτελεί το σύμβολο με το μεγαλύτερο κέρδος.

- Κενά Σύμβολα

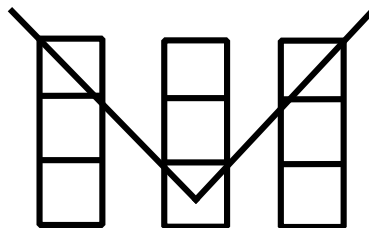
Υπάρχει η περίπτωση θέσεις των τροχών να είναι κενές. Συνήθως όταν χρησιμοποιούνται τα κενά παίρνουν τις θέσεις πριν και μετά από εκείνες των υπόλοιπων συμβόλων που υπάρχουν.

Κάθε τροχός θα πρέπει τουλάχιστον να αποτελείται από βασικά σύμβολα. Οι άλλες κατηγορίες χρησιμοποιούνται για να κάνουν το παιχνίδι πιο ενδιαφέρον και διασκεδαστικό.

4. pay lines : Αποτελούν την σειρά θέσεων των τροχών για την οποία θα γίνει έλεγχος αν ο συνδυασμός που εμφανίζεται κερδίζει ή όχι. Για παράδειγμα για 3 τροχούς και 3 γραμμές μπορεί να κερδίζει η μεσαία.



Δεν είναι απαραίτητο να είναι στην ίδια ευθεία. Για το ίδιο παράδειγμα φαίνεται παρακάτω μία pay line που ακολουθεί διαφορετική σειρά θέσεων.



Κάθε μηχανή slot έχει τουλάχιστον μία pay line. Παλαιότερα υπήρχε μόνο μία σε όλες τις μηχανές. Σήμερα υπάρχουν

σύγχρονες μηχανές slot που περιέχουν παραπάνω παραπάνω και από 100 pay lines.

5. Ποσό στοιχήματος (Bet)

Το ποσό που στοιχηματίζεται σε κάθε γυρίσματα διαφέρει ανάλογα τη μηχανή slot και τον αριθμό των paylines που έχει επιλεχθεί. Αν για παράδειγμα για μία payline πρέπει να στοιχηματιστεί 1 νόμισμα τότε για πέντε το ποσό θα είναι 5 νομίσματα. Αυτό συμβαίνει για να υπάρχει ισορροπία στο παιχνίδι. Όπως είναι λογικό αφού αυξάνονται οι πιθανότητες να κερδίσει κάποιος με περισσότερες από μία paylines το ποσό που χαθεί αν δεν υπάρξει σωστός συνδυασμός θα πρέπει να είναι και αυτό μεγαλύτερο.

6. Πίνακας νικητήριων συνδυασμός (pay table)

Αυτός ο πίνακας περιλαμβάνει όλους τους συνδυασμούς και τα πόσα που θα κερδίσει ο παίχτης αν κάποιος από αυτούς εμφανιστεί στην οθόνη

Γενικά οι παραλλαγές μίας μηχανής slot είναι πολλές και ποικίλουν ανάλογα με τις επιθυμίες κάθε κατασκευαστικής εταιρίας ή του προγραμματιστή. Όλες όμως πρέπει να έχουν τα παραπάνω χαρακτηριστικά.

4.3 Περιγραφή Εφαρμογής Slot Machine

Όπως όλες οι μηχανές slot έτσι και αυτή που υλοποιήθηκε στην παρούσα εργασία περιλαμβάνει όλα τα χαρακτηριστικά που αναφέρθηκαν παραπάνω. Σε αυτή την παράγραφο θα παρουσιαστούν αναλυτικά αυτά τα χαρακτηριστικά της εφαρμογής.

Αρχικά όσον αφορά τους τροχούς επιλέχθηκε αυτοί να είναι 5 με τον καθένα απο αυτούς να περιλαμβάνει 22 θέσεις συμβόλων. Τα σύμβολα που τοποθετήθηκαν σε κάθε τροχό είναι τα εξής :

- Βασικά Σύμβολα



- "wild" Σύμβολο



- "Jackpot" Σύμβολο



Σε κάθε τροχό υπάρχουν 4 x , 4 x , 3 x , 3 x , 2 x , 2 x , 2 x , 1 x , 1 x  να συμπληρώνουν τις 22 θέσεις ανακατεμένα.

Επίσης υποστηρίζει τη δυνατότητα ο παίχτης να επιλέξει ανάμεσα σε μία ,τρεις ή πέντε pay lines για κάθε γύρισμα των τροχών. Για κάθε αλλαγή των pay lines αλλάζει και το ποσό που στοιχηματίζεται κάθε φορά. Για μία pay line στοιχηματίζεται 1 νόμισμα, για τρεις στοιχηματίζονται 3 νομίσματα και για πέντε pay lines 5 νομίσματα. Τέλος για να υπάρξή επιτυχής συνδυασμός θα πρέπει το ίδιο σύμβολο να εμφανίζεται 3, 4 ή 5 φορές στην σειρά ξεκινώντας από τα αριστερά. Τα ποσά που θα κερδίσει ο παίχτης για κάθε επιτυχή συνδυασμό φαίνονται στον πίνακα που ακολουθεί.

Σύμβολα	3 εμφανίσεις	4 εμφανίσεις	5 εμφανίσεις
	5	15	50
	5	15	50
	5	15	50
	10	25	60
	10	25	75
	15	50	100
	20	60	150
	75	200	500
	250	500	2500

Πίνακας 4.1: Τα ποσά για κάθε επιτυχή συνδυασμό της slot machine

Το σύμβολο  αν εμφανιστεί συνεχόμενες φορές από την πρώτη θέση στα αριστερά τότε αντιμετωπίζεται ως βασικό σύμβολο και δεν κάνει χρήση της ιδιότητας του σαν "wild". Αντίθετα στις περιπτώσεις όπου

συνδυάζεται με ένα άλλο σύμβολο εκτός του  και το "wild" δεν κατέχει 3 ή 4 θέσεις συνεχόμενες από τα αριστερά, τότε κάνει χρήση της ιδιότητας και αντιμετωπίζεται ως το δεύτερο σύμβολο με το οποίο συνδυάζεται.

4.4 Συνδυασμοί Και Πιθανότητες

Σχετικά με τους πιθανούς συνδυασμούς που εμφανίζονται στη συγκεκριμένη slot machine με τροχούς 22 θέσεων, υπάρχουν συνολικά $22 \cdot 22 \cdot 22 \cdot 22 \cdot 22 = 5153632$ διαφορετικές εμφανίσεις. Οι συνδυασμοί που κερδίζουν χωρίζονται σε δύο κατηγορίες. Στη πρώτη κατηγορία ανήκουν οι εκείνοι όπου ένα σύμβολο εμφανίζεται 3, 4 ή 5 φορές συνεχόμενα από τα αριστερά. Στο πίνακα που ακολουθεί φαίνονται αυτοί οι πιθανοί συνδυασμοί με την συχνότητα και την πιθανότητα εμφάνισης για τον καθένα:

Σύμβολο	Συχνότητα	Πιθανότητα (%)
   x x	17280	0.335298
    x	3840	0.074511
    	1024	0.019869
   x x	17280	0.335298
    x	3840	0.074511
    	1024	0.019869
   x x	8208	0.159266
    x	1296	0.025147
    	243	0.004715
   x x	2720	0.052778
    x	272	0.005278

	32	0.000621
	2720	0.052778
	272	0.005278
	32	0.000621
	2720	0.052778
	272	0.005278
	32	0.000621
	378	0.007335
	18	0.000346
	1	0.000019
	9747	0.189129
	1539	0.029862
	243	0.004715
	441	0.008557
	21	0.000407
	1	0.000019

Πίνακας 4.2: Οι συνδιασμοί και πιθανότητες εμφάνισης τους

Όπου x οποιοδήποτε άλλο σύμβολο.

Στην δεύτερη κατηγορία ανοίκουν οι συνδυασμοί οι οποίοι περιέχουν ένα οποιοδήποτε σύμβολο (εξτός του ) και το "wild"  σύμβολο. Ο υπολογισμός της συχνότητας και πιθανότητας εμφάνισης τους γίνεται όπως και στον παραπάνω πίνακα ως εξής:

Έστω ότι ζητάμε την συχνότητα εμφάνισης του συνδυασμού



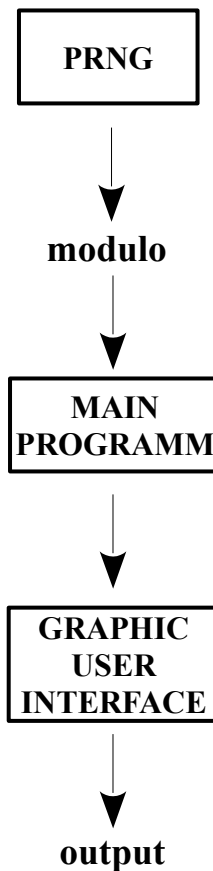
τότε θα έχουμε $2 \cdot 3 \cdot 2 \cdot (22 - 2 - 3) \cdot (22 - 2) = 4080$ πιθανές εμφανίσεις (Στη θέση 4 αφαιρείται το 3 καθώς δεν μπορεί να είναι ).

Και η πιθανότητα εμφάνισης θα είναι $(4080 / 5153632) \cdot 100 = 0,079167\%$.

Το συνολικό πλήθος των συνδυασμών που κερδίζουν είναι ίσο με 75496 επομένως η πιθανότητα να προκύψει ένα από αυτούς τους συνδυασμούς ισούται με $(75496 / 5153632) \cdot 100 = 1,46491\%$. Αυτή η πιθανότητα μπορεί να αυξηθεί αν αυξηθούν και οι pay lines που χρησιμοποιούνται κάθε φορά. Για 3 pay lines η πιθανότητα γίνεται $3 \cdot 1,46491 = 4,39473$.

4.5 Υλοποίηση Εφαρμογής

Μία μηχανή slot δεν είναι τίποτα άλλο από έναν πρόγραμμα που έχει σχεδιαστεί έτσι ώστε να δίνει λιγότερα λεφτά από ότι δέχεται. Η κατασκευή της είναι απλή. Αποτελείται από μία ψευδοτυχαία γεννήτρια παραγωγής αριθμών, μία εικονική διεπαφή ώστε να γίνεται η επικοινωνία με τον παίχτη και ένα πρόγραμμα που συνδέει την γεννήτρια με την διεπαφή όπου εκτελούνται όλοι οι έλεγχοι, οι διαδικασίες και οι υπολογισμοί ώστε να εμφανίζεται το σωστό αποτέλεσμα. Η βασική της δομή φαίνεται στο παρακάτω σχήμα.



Σχήμα 4.1: Η βασική δομή μία μηχανής slot

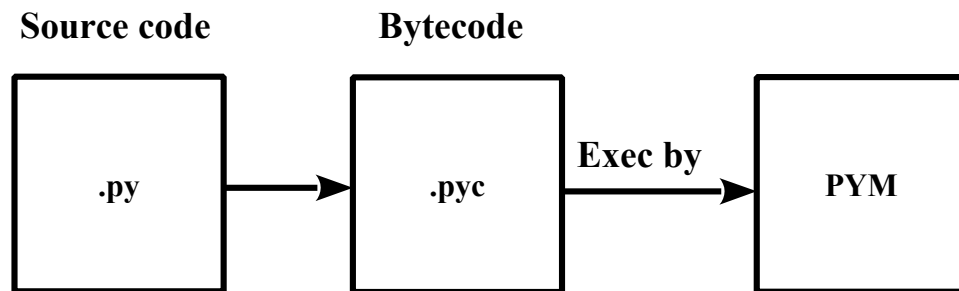
4.5.1 Η Γλώσσα Python

Πριν προχωρήσουμε στην παρουσίαση της υλοποίησης της μηχανής slot που περιγράφηκε στην προηγούμενη παράγραφο κρίθηκε απαραίτητο να αφιερώσουμε λίγο χρόνο για την γλώσσα προγραμματισμού που επιλέχθηκε για την υλοποίηση και τα εργαλεία που χρησιμοποιήθηκαν.

Η γλώσσα προγραμματισμού που επιλέχθηκε είναι η Python και αυτή η επιλογή δεν έγινε καθόλου τυχαία. Η Python είναι μία γλώσσα που τα τελευταία χρόνια έχει επικρατήσει ανάμεσα στις πιο γνωστές και

περιζήτητες γλώσσες. Δημιουργήθηκε από τον ολλανδό Guido van Rossum και πήρε το όνομα της από το αγαπημένο του τηλεοπτικό θέαμα, το ιπτάμενο τσίρκο των Mothy Python's. Η ανάπτυξη της ξεκίνησε το 1989 και μέχρι σήμερα εξελίσσεται διαρκώς.

Αποτελεί μία διερμηνευόμενη γλώσσα δηλαδή υπάρχει ένα πρόγραμμα "ο διερμηνέας" που εκτελεί τις γραμμές κώδικα την μία μετά την άλλη. Ότι είναι μία διερμηνευόμενη γλώσσα σημαίνει ότι δεν υπάρχει ενδιάμεση μορφή μεταφρασμένου κώδικα. Αυτό δεν είναι απόλυτα αληθές για την Python. Σε αυτή όπως και σε άλλες γλώσσες δημιουργείται ένας ενδιάμεσος κώδικας (μεταγλωττισμένης μορφής) που θα την δούμε με το όνομα Bytecode. Και στην συνέχεια ένα πρόγραμμα με όνομα Python Virtual Machine (PYM) αναλαμβάνει να εκτελέσει το Bytecode.



Σχήμα 4.2: Αναπαράσταση της διαδικασίας εκτέλεσης κώδικα σε Python

Επίσης είναι μία αλληλεπιδραστική και αντικειμενοστραφής γλώσσα.

Τα κύρια χαρακτηριστικά της είναι

1. Είναι γλώσσα υψηλού επιπέδου.
2. Είναι μία γλώσσα γενικού σκοπού, δηλαδή δεν είναι εστιασμένη σε συγκεκριμένο τομέα.

3. Χαρακτηρίζεται από την αναγνωρισιμότητα και την απλότητα της. Η αναλογία κώδικα ανάμεσα στην Python και άλλες γλώσσες είναι 1 προς 5.
4. Αποτελεί ελεύθερη ανοιχτού κώδικα γλώσσα.
5. Μπορεί να χρησιμοποιήσει κομμάτια κώδικα γραμμένα σε άλλες γλώσσες, για να εξασφαλίσει μεγαλύτερη ταχύτητα.
6. Υποστηρίζει όλες τις ιδιότητες του αντικειμενοστραφούς προγραμματισμού.
7. Διαθέτει βιβλιοθήκες, που βελτιώνονται συνεχώς και καλύπτουν το μεγαλύτερο εύρος επιστημονικών πεδίων.

Όλα τα παραπάνω χαρακτηριστικά καθόρισαν την επιλογή της για την παρούσα υλοποίηση. Η έκδοση της Python που χρησιμοποιήθηκε είναι η 3.6 και η σύνταξη της έγινε στο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) PyCharm που έχει αναπτυχθεί από την εταιρεία JetBrains.

```
16 File Edit View Shell Tools Refactor Run Debug Window Help
Std Machine -> idmachinery.py
Project -> idmachinery.py
idmachinery.py
17 temp.append(self.activateLine(i))
18 temp.append(self.activateLine(i))
19 line_combs.append(copy.copy(temp))
20
21 for index, combo in enumerate(line_combs):
22     counter = 0
23     win_symbol = ""
24
25     for i in range(9, 51):
26         if combo[i] == "0":
27             counter += 1
28         else:
29             if win_symbol == "":
30                 win_symbol = combo[i]
31                 counter += 1
32             elif win_symbol == combo[i]:
33                 counter += 1
34             elif win_symbol != combo[i]:
35                 break
36
37     if counter >= 3:
38         self.winning_lines.append(index)
39
40     if win_symbol == "":
41         if counter == 3:
42             print("Win x" + str(counter) + " symbol " + win_symbol)
43             self.win += 25
44             elif counter == 4:
45                 print("Win x" + str(counter) + " symbol " + win_symbol)
46                 self.win += 200
47             elif counter == 5:
48                 print("Win x" + str(counter) + " symbol " + win_symbol)
49                 self.win += 500
50             elif win_symbol == "0":
51                 if counter == 5:
52                     print("Win x" + str(counter) + " symbol " + win_symbol)
53                     self.win += 20
54                     elif counter == 6:
55                         print("Win x" + str(counter) + " symbol " + win_symbol)
56                         self.win += 60
57                     elif counter == 7:
58                         print("Win x" + str(counter) + " symbol " + win_symbol)
59
60         elif counter == 9:
61             print("Win x" + str(counter) + " symbol " + win_symbol)
```

4.5.2 Υλοποίηση Mersenne Twister

Η γεννήτρια που χρησιμοποιήθηκε στην κατασκευή της εφαρμογής slot machine είναι η MT19937. Υλοποιήθηκε σε μία κλάση με βάση την περιγραφή που παρουσιάστηκε στο Κεφάλαιο 3. Παρακάτω φαίνεται ο κώδικας αυτής με όνομα MersenneTwister.

```
class MersenneTwister:

    def __init__(self):
        self.w = 32
        self.n = 624
        self.m = 397
        self.r = 31
        self.f = 1812433253
        self.a = 0x9908B0DF
        self.u, self.d = 11, 0xFFFFFFFF
        self.s, self.b = 7, 0x9D2C5680
        self.t, self.c = 15, 0xEFC60000
        self.l = 18

        self.states = [0 for i in range(self.n)]
        self.index = 0

    def initialize(self, seed=5489):
        self.states[0] = seed
        self.index = self.n
        for i in range(1, self.n):
            temp = self.f * (self.states[i - 1]^(self.states[i - 1] >> self.w - 2))) + i
            self.states[i] = temp & 0xFFFFFFFF

    def random_int(self):
        if self.index >= self.n:
            self.twist()
            self.index = 0

        y = self.states[self.index]
        y = y ^ ((y >> self.u) & self.d)
        y = y ^ ((y << self.t) & self.c)
        y = y ^ ((y << self.s) & self.b)
        output = y ^ (y >> self.l)

        self.index += 1
        return output & 0xFFFFFFFF

    def twist(self):
        for i in range(0, self.n):
            x=(self.states[i] & 0x80000000)+(self.states[(i + 1) % self.n] & 0x7FFFFFFF)
            xA = x >> 1
            if x % 2 != 0:
                xA = xA ^ self.a
            self.states[i] = self.states[(i + self.m) % self.n] ^ xA
```

Εδώ να σημειωθεί πως αν δεν δοθεί κάποιος σπόρος στην γεννήτρια αυτός παίρνει την τιμή 5489 που χρησιμοποιείται και στην ANSI C.

4.5.3 Υλοποίηση κλάσης Slot machine

Η δεύτερη κλάση που υλοποιήθηκε ονομάστηκε SlotMachine και σε αυτή συμβαίνει η εξαγωγή ενός συνδυασμού συμβόλων, ο έλεγχος αν ο συνδυασμός αυτός κερδίζει και τέλος ο υπολογισμός του διαθέσιμου ποσού νομίσματος που διαθέτει ο παίχτης μετά από κάθε γύρισμα των τροχών.

Ας δούμε τώρα πιο αναλυτικά τι συμβαίνει σε αυτή την κλάση. Αρχικά η κλάση συνδέεται με την κλάση MersenneTwister της γεννήτριας με χρήση της εντολής `from mersenne_twister import MersenneTwister` και με την βιβλιοθήκη `from datetime import datetime`. Στην συνέχεια υλοποιήθηκαν τρεις μέθοδοι η `__init__` , `get_combination` και `check_if_win`.

Η μέθοδος `__init__` αποτελεί τον κατασκευαστή του αντικειμένου. Σε αυτή αρχικοποιούνται 5 πίνακες οι οποίοι περιγράφουν τους τροχούς της μηχανής slot καθώς επίσης όποια άλλη μεταβλητή είναι απαραίτητη για την σωστή εκτέλεση του κώδικα όπως είναι το ποσό που διαθέτει ο παίχτης, με πόσες paylines παίζει και άλλες. Τέλος κατασκευάζει ένα αντικείμενο MersenneTwister και αρχικοποιεί την γεννήτρια κάνοντας χρήση ενός σπόρου. Ο σπόρος δημιουργείται με συνδυασμό του μήνα, ημέρας, χρονιάς, ώρας, λεπτού και δευτερολέπτου την στιγμή που εκτελέστηκε το πρόγραμμα.

Στη επόμενη μέθοδο `get_combination` παράγονται 5 τυχαίοι αριθμοί καλώντας την μέθοδο `random_int` του αντικειμένου MersenneTwister που

δημιουργήθηκε στην `__init__` . Σε κάθε ένα από αυτούς τους ακέραιους αριθμούς πραγματοποιείται η πράξη `mod 22` και στην συνέχεια τοποθετούνται σε ένα πίνακα ο οποίος αντιπροσωπεύει τον συνδυασμό της slot machine για εκείνο το γύρισμα. Ο λόγος που συμβαίνει αυτή η πράξη `mod 22` δεν είναι από το να μετατραπούν οι ακέραιοι σε αριθμούς μεταξύ του 0 και 22 που αποτελούν τις θέσεις των συμβόλων στους τροχούς.

Τέλος η `check_if_win` κάνοντας χρήση κάποιων βροχών `for` και εντολών `if` ελέγχει αν ο συνδυασμός ή οι συνδυασμοί (αν οι `paylines` είναι περισσότερες από μία) που δημιουργήθηκαν κερδίζουν ή όχι . Αν ο παίχτης κέρδισε υπολογίζει το ποσό των νομισμάτων που αναλογεί στον συνδυασμό.

4.5.4 Υλοποίηση Διεπαφής

Για να εμφανίζεται το αποτέλεσμα στην οθόνη και για να εκτελεί ο παίχτης όλες τις λειτουργίες που προσφέρει η εφαρμογή δημιουργήθηκε μία γραφική διεπαφή κάνοντας χρήση της βιβλιοθήκης `pygame` (v1.9.6) της Python. Η `pygame` είναι μία βιβλιοθήκη της Python που έχει δημιουργηθεί με σκοπό τον προγραμματισμό παιχνιδιών.

Αρχικά αρχικοποιείται η `pygame` για όλες τις κλάσεις που την αποτελούν με την εντολή `pygame.init`. Αν δεν χρησιμοποιηθεί αυτή εντολή θα πρέπει να γίνουν `import` μία προς μία όλες οι κλάσεις που απαιτούνται. Στην συνέχεια κατασκευάζεται ένα παράθυρο μεγέθους 950 x 730 pixels χρησιμοποιώντας τη μέθοδο `pygame.display.set_mode`. Στη μέθοδο `redraw_screen` που υλοποιήθηκε σχεδιάζεται η εμφάνιση της διεπαφής κάνοντας χρήση μεθόδους της κλάσης `draw` της `pygame` για τον σχεδιασμό

βασικών σχημάτων , της image για την φόρτωση διάφορων εικόνων από τον φάκελο του προγράμματος και την display ώστε να τοποθετείται το κάθε σχήμα ή εικόνα στο επιθυμητό σημείο στο παράθυρο. Παρακάτω ακολουθούν η βασική απεικόνιση της εφαρμογής και εκείνη του pay table



Η `redraw_screen` καλείται μετά από κάθε ενέργεια ώστε να εμφανίζεται το αποτέλεσμα στην οθόνη.

Τέλος όσο αναφορά την αλληλεπίδραση με τον παίχτη αυτή υλοποιήθηκε κάνοντας χρήση των γεγονότων του ποντικιού (mouse events). Κάθε ένα από τα κουμπιά "spin", "pay table", "+", "-" της διεπαφής καλύπτουν συγκεκριμένα pixels στο παράθυρο. Όταν πατηθεί το ποντίκι στην οθόνη το πρόγραμμα ελέγχει αν αυτό το σημείο είναι στο ανάμεσα στο εύρος των pixels που καλύπτει κάθε κουμπί. Αν αυτό είναι αληθές τότε πραγματοποιείται η λειτουργία που έχει αντιστοιχηθεί σε αυτό το κουμπί.

Κατάλογος Σχημάτων / Πινάκων

Σχήμα 1.1: Αναπαράσταση της ταξινόμησης των RNGs	17
Σχήμα 1.2: Παράδειγμα Γεννήτριας Ψευδοτυχαιότητας.....	17
Σχήμα 1.3: Παράδειγμα Γεννήτριας Ψευδοτυχαιότητας με τυχαίο σπόρο.....	18
Πίνακας 1.1: Χαρακτηριστικά των PRNGs και TRNGs.....	20
Σχήμα 2.1: Αναπαράσταση κύκλου μιας PRNG.....	24
Πίνακας 2.1: Αποτελέσματα δέκα επαναλήψεων της $x_n = (7x_{n-1}) \bmod 32$	31
Πίνακας 2.2: Αποτελέσματα δέκα επαναλήψεων της $x_n = (5x_{n-1}) \bmod 32$	32
Πίνακας 2.3: Αποτελέσματα δέκα επαναλήψεων μίας BBS για $m=192649$ και $s = 101355$	39
Πίνακας 2.4: Αποτελέσματα δεκαπέντε επαναλήψεων μίας LFSR για $k=4$, $r = 1$ και σπόρο $[1,0,1,0]$	47
Σχήμα 2.2: Αναπαράσταση μίας LFSR με χρήση καταχωρητή ολίσθησης.....	48
Πίνακας 3.1: Παράμετροι της γεννήτριας MT19937.....	57
Πίνακας 4.1: Τα ποσά για κάθε επιτυχή συνδιασμό της slot machine.....	67
Πίνακας 4.2: Οι συνδιασμοί και πιθανότητες εμφάνισης τους.....	68
Σχήμα 4.1: Η βασική δομή μία μηχανής slot.....	71
Σχήμα 4.2: Αναπαράσταση της διαδικασίας εκτέλεσης κώδικα σε Python.....	72

Βιβλιογραφία

- [1] Κρυπτογραφία & Ασφάλεια δικτύων, William Stallings, Πρώτη ελληνική έκδοση
- [2] Σύγχρονη κρυπτογραφία Θεωρία και εφαρμογές, Mike Burmester, Στέφανος Γκριτζάλης
- [3] Μια υπολογιστική εισαγωγή στη θεωρία αριθμών, Victor Shoup
- [4] Αριθμητική ανάλυση, Α' τόμος, Αβδέλα Γεώργιου, Σίμου Θεόδωρου
- [5] www.random.org
- [6] www.en.wikipedia.org/wiki/Applications_of_randomness
- [7] www.en.wikipedia.org/wiki/Hardware_random_number_generator
- [8] Random Number Generation: Types and Techniques, David DiCarlo
- [9] www.courses.grainger.illinois.edu/phys466/fa2020/lnotes/random_numbers.html
- [10] www.geeksforgeeks.org/pseudo-random-number-generator-prng/
- [11] www.cryptosys.net/rng_algorithms.html
- [12] <https://towardsdatascience.com/building-a-pseudorandom-number-generator-9bc37d3a87d5>
- [13] A Search for Good Pseudo-random Number Generators : Survey and Empirical Studies, Kamalika Bhattacharjee
- [14] On a new class of pseudorandom number for simulation methods, Harald Niederreiter
- [15] Multiple Recursive Random Number Generators and their APL programmes, Jean-Louis Foulley
- [16] A new class of linear feedback shift register generators, Pierre L'Ecuyer, Francois Panneton
- [17] Pseudorandom numbers for mini- and microcomputers: A generalized

- feedback shift register algorithm, W. H. Payne
- [18] Twisted GFSR Generators ,Makoto Matsumoto
 - [19] Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator, Makoto Matsumoto
 - [20] Mersenne Twister Random Number Generation on FPGA, CPU and GPU , Xiang Tian and Khaled Benkrid
 - [21] www.en.wikipedia.org/wiki/Mersenne_Twister
 - [22] www.sciencedirect.com/topics/computer-science/mersenne-twister
 - [23] www.dl.acm.org/doi/10.1145/272991.272995
 - [24] www.math.sci.hiroshima-u.ac.jp/m-mat/MT/emt.html
 - [25] PAR Sheets, probabilities, and slot machine play:Implications for problem and non-problem gambling, Kevin A. Harrigan and Mike Dixon
 - [26] Addiction by Design: Machine Gambling in Las Vegas, Natasha Dow Shull
 - [27] Do Weighted Reels on a Slot Machine Distort a Gambler's Judgment of Probability?The Effect of Near Misses, Keith D. Horton
 - [28] www.easy.vegas/games/slots/how-they-work
 - [29] A Personal Guide to Electronic Slot Machines, John M. Pacyniak
 - [30] Slot Machines: Pursuing Responsible Gaming Practices for Virtual Reels and Near Misses, Kevin A. Harrigan