هفتمین کنفرانس بین المللی انجمن رمز ایران

۲۴ و ۲۵ شهریورماه ۱۳۸۹، دانشگاه صنعتی خواجه نصیرالدین طوسی

انجمن رمزایران

دانشگاه صنعتی خواجه نصیرالدین طوسی

سامانه میانضربی آشوبگونه برای تولید اعداد شبه تصادفی

مجید بابایی[۱]، حامد رحیم‌اف[۲]، محسن فرهادی[۳]، محمد رضا جاهد مطلق[۴]

دانشکده کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی شاهرود[۱]

babaee@comp.tup.ac.ir

دانشکده کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی شاهرود[۲و۳]

دانشکده کامپیوتر و فناوری اطلاعات، دانشگاه علم و صنعت[۴]

چکیده

مولدهای شبه تصادفی از اهمیت ویژه‌ای در رمزنگاری به خصوص جهت تولید دنباله کلید اجرایی در سیستم‌های رمز پی‌درپی برخوردارند. سرعت تولید و پراکندگی بالا از ویژگی‌های مورد علاقه دانشمندان در تولید اعداد شبه تصادفی است که با کشف پدیده آشوب و معادلات حاکم بر آن وارد مرحله جدیدی گشته است. در حقیقت ورود معادلات آشوب در مولدهای شبه تصادفی باعث به وجود آمدن حساسیت بسیار زیاد مولدها به مقدارهای اولیه شده است. یکی از روش‌های کلاسیک تولید اعداد شبه تصادفی روش میانضربی است که در عین سادگی با سرعت بالا و پراکندگی مناسبی داده‌های تصادفی را تولید می‌کند. ولی علی‌رغم این ویژگی‌های منحصر بفرد دارای نقاط ضعفی است که کاربرد وسیع‌تر این الگوریتم را با مشکل روبرو کرده است. در این مقاله بعد از معرفی تعدادی از مولدهای شبه تصادفی به بررسی الگوریتم میانضربی پرداخته شده است و در ادامه نگاشت آشوبگونه لوجستیک به همراه ویژگی‌های خاص آن معرفی و با کمک آن ضعف الگوریتم میانضربی در همگرایی زودرس و تولید تعداد محدود عدد شبه تصادفی برطرف شده است. در نهایت با انجام شبیه ساز تست مونت کارلو بهینگی روش پیشنهادی نسبت به روش میانضربی اثبات شده‌است.

واژه‌های کلیدی

رمزنگاری، تولید کننده اعداد شبه تصادفی، روش میانضربی، تابع آشوبگونه لوجستیک، تست مونت کارلو.

۱- مقدمه

رمزنگاری از دیرباز به عنوان یک ضرورت برای حفاظت از اطلاعات خصوصی در مقابل دسترسی‌های غیرمجاز در تجارت، سیاست و مسایل نظامی وجود داشته‌است. تولید اعداد شبه تصادفی یکی از مباحث کاربردی در علم رمزنگاری به شمار می‌رود که دارای الگوریتم‌های بسیاری در علوم کامپیوتر است [1-4].

دنباله‌های تصادفی براساس دو روش کلی پیاده‌سازی می‌شوند، در روش اول تولید اعداد تصادفی براساس ویژگی اختلاف‌کم[1] است که با کمک الگوریتم‌های بسیار پیچیده ریاضی سعی در تولید داده‌هایی با حداکثر فاصله در فضای چند بعدی دارد. در روش دوم اعداد تصادفی براساس ویژگی اختلاف زیاد[2] تولید می‌شوند. هر کدام از روش‌های اشاره شده مزایا و معایب خود را دارد که در ادامه نگاهی گذرا به این مفاهیم خواهیم داشت.

در ابتدا تمرکز دانشمندان بر روی مولدهای تصادفی بود که بتوانند با دنباله‌های تولید شده تمام فضای حالت را بپوشانند، یعنی داده‌ها از پراکندگی قابل قبولی برخوردار باشند این دنباله ها به نام دنباله‌های دارای ویژگی اختلاف کم شهرت یافت.

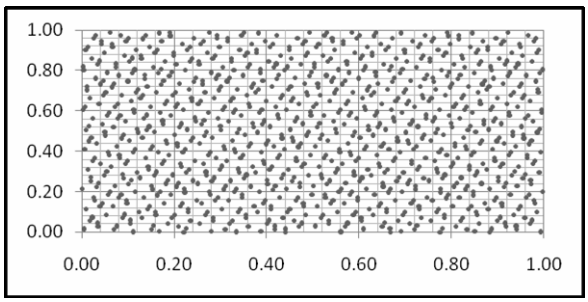
دنباله Corput اولین دنباله‌ای بود که بر این اساس شکل گرفت، اما داده‌های تصادفی تولید شده از این دنباله در ابعاد بالا، حالت تصادفی خود را از دست داده و از یک تابع خطی تبعیت می‌کند. پس از آن الگوریتم هالتون[3] در سال ۱۹۶۰ ارایه شد، در

[1] Low discrepancy

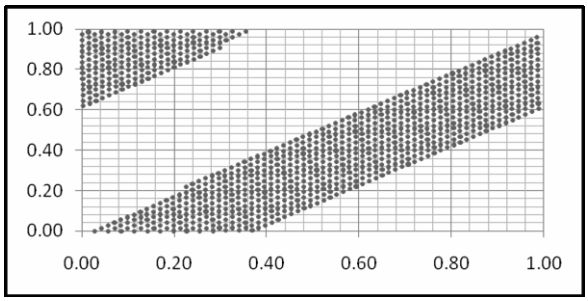
[2] High discrepancy

[3] Halton

تعریف کوتاهی از دنباله هالتون باید گفت، دنباله هالتون همان دنباله Corput، با مقدار پایه‌ای برابر با n امین عدد اول برای n امین بعد دنباله هالتون است. همانطور که در شکل ۱ مشاهده می‌شود، اعداد تولید شده در بعد بیستم دنباله هالتون دارای پراکندگی مناسبی هستند ولی با بالا رفتن بعد این همگرایی نمود بیشتری پیدا می‌کند، تا آنجا که با توجه به شکل ۲ ، در بعد ۴۲۰ روند تصادفی خود را از دست داده است. به هرحال، دنباله هالتون همگرایی کمتری نسبت به دنباله Corput در ابعاد بالاتر دارد [5].

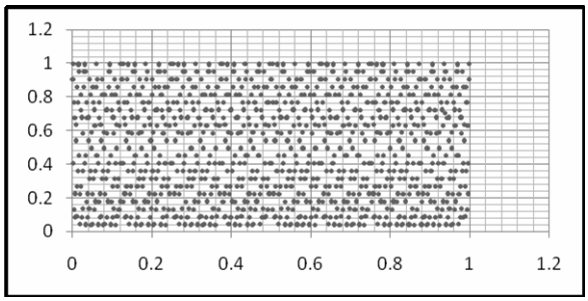


شکل ۱- اعداد تصادفی تولید شده در بعد ۲۰ دنباله هالتون

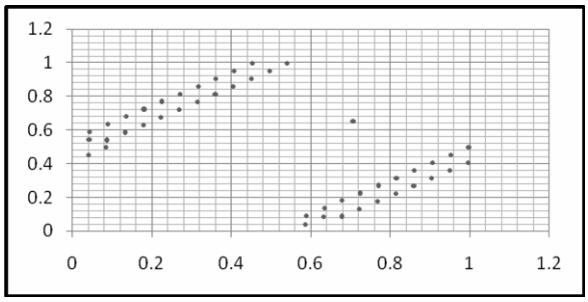


شکل ۲- اعداد تصادفی تولید شده در بعد ۴۲۰ دنباله هالتون

دنباله فیوره[1] یک دنباله چند بعدی است که بعد از دنباله هالتون در سال ۱۹۶۲ ارایه شد. با توجه به الگوریتم این دنباله، مقدار پایه همان کوچکترین عدد اولی است که بزرگتر یا مساوی شماره بعد در مسأله است، (برای بعد اول از پایه با مقدار ۲ استفاده می‌شود)، به عنوان مثال، دنباله هالتون در بعد ۵۰، پنجاهمین عدد اول را که ۲۲۹ است را به عنوان طول حلقه در نظر می‌گیرد که بسیار بزرگ است ولی در دنباله فیوره عدد ۵۳ (که اولین عدد اول بزرگتر از مقدار بعد است) برای اندازه حلقه در نظر گرفته می‌شود. بنابراین دنباله فیوره در ابعاد بزرگ سریع تر از هالتون است [5]. اما همچنان مشکل همگرایی دنباله در ابعاد بالاتر وجود دارد، همانطور که در شکل‌های ۳ و ۴ مشاهده می-کنید با بالا رفتن بعد پراکندگی اعداد تولید شده توسط دنباله کاهش یافته است.

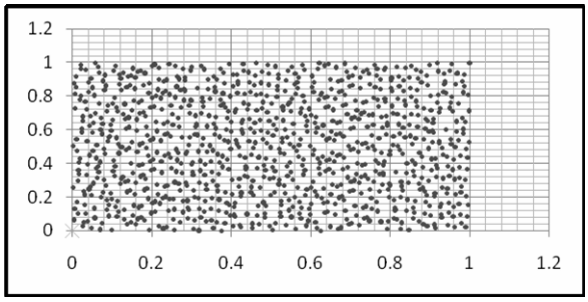


شکل ۳- اعداد تصادفی تولید شده در بعد ۲۰ دنباله فیوره

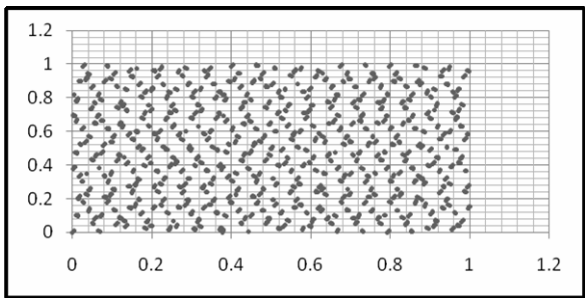


شکل ۴- اعداد تصادفی تولید شده در بعد ۴۲۰ دنباله فیوره

در سال ۱۹۶۷ دنباله سبل[2] با شباهت‌های بسیار به دنباله فیوره ارایه شد، در دنباله فیوره هر بعد از یک المان برداری متفاوت بدست می‌آمد در صورتی که در دنباله سبل از مقدار ثابت پایه دو برای تمام ابعاد استفاده می‌شود، بنابراین دنباله سبل بسیار سریع‌تر و ساده‌تر است. که این ویژگی منحصر بفرد دنباله سبل سبب تولید اعداد تصادفی با همگرایی کمتر در ابعاد بالا با سرعت بیشتر می‌شود که در شکل های ۵ و ۶ مشخص است.



شکل ۵- اعداد تصادفی تولید شده در بعد ۲۰ دنباله سبل



شکل ۶- اعداد تصادفی تولید شده در بعد ۴۲۰ دنباله سبل

[1] Faure

[2] Sobo

در روش‌های اشاره شده، از الگوریتم‌های بسیار پیچیده با حلقه های بلند برای تولید اعداد تصادفی استفاده می‌شود، اما در بسیاری از شرایط ما نیاز به الگوریتمی داریم که اعداد تصادفی را با سرعت بالا، پراکندگی مناسب و تعداد کافی در اختیار ما قرار دهد.

در ادامه به معرفی دنباله‌های با ویژگی اختلاف‌زیاد می‌پردازیم که از جهات مختلف نسبت به دنباله‌های پیشین برتری دارند، مهم ترین مزیت استفاده از این دنباله‌ها علاوه بر پراکندگی بالای اعداد تولید شده، پیچیدگی زمانی مناسب است.

دنباله LCG یکی از دنباله‌های با ویژگی اختلاف زیاد است، که توسط تابعی بازگشتی در مقدار اولیه‌ای خاص اعداد تصادفی مناسبی را در زمان قابل قبول تولید می‌کند [4]. از دیگر مولدهای تصادفی با ویژگی اختلاف زیاد می‌توان به روش میان مربع، مضرب ثابت و میانضربی اشاره کرد.

در سال‌های اخیر با ورود توابع آشوب به مولدهای تصادفی شاهد رشد چشمگیری در توسعه سیستم‌های تصادفی هستیم که از آن جمله می‌توان به مولد تصادفی Kolesov در سال ۲۰۰۱ اشاره کرد که اعداد تصادفی را براساس سیگنال‌های گسسته آشوبگونه تولید می‌کرد [5]. مدتی بعد با کمک یک سیستم آشوبگونه تکه‌ای خطی[3] یک مولد بسیار مناسب اعداد شبه تصادفی طراحی شد [6]. همچنین در سال ۲۰۰۳، Jakimoski و همکاران امکانات متفاوتی از آشوب را برای تولید اعداد تصادفی ارایه کرد [7]. در سال ۲۰۰۴ مولدی بر اساس الگوریتم آشوبگونه تکه‌ای خطی ساخته شد که در یک مسیر واحد، سیستم‌های آشوبگونه به هم متصل می‌شدند و نشان داده شد، با داشتن عمکرد مناسبی از این سیستم‌ها دنباله بسیار مناسبی از اعداد تصادفی تولید می‌شوند، که دارای دوره تناوب بلند و سرعت تولید بالا است [8]. در سال ۲۰۰۵ یک ایده جدید از مولدهای تصادفی آشوبگونه صورت گرفت که بر اساس مدل اصلاح شده سیستم‌های قبلی بود [9]. پس از آن یک مولد اعداد تصادفی آشوبگونه توسط Wang و همکاران طراحی شد که براساس یک مدار آنالوگ بود [10]. در سال ۲۰۰۶ نیز Wang پیشنهاد تولید اعداد تصادفی را بر اساس یک z-logistic عنوان کرد به طوری که دنباله بیت‌های تصادفی براساس یک مدار آشوبگونه با دقت محاسباتی محدود تولید شوند [11]. در سال ۲۰۰۷ نیز Ergun و همکارانش نشان دادند که دنباله بیت‌ها می‌تواند از یک مدار الکترونیکی غیر مستقل آشوبگونه تولید شود [12]. در سال ۲۰۰۹ یک مولد اعداد تصادفی حقیقی بر اساس حرکت موس ارایه شد که قادر بود ۲۵۶ بیت تصادفی را با سرعتی مناسب تولید کند [13].

با توجه به این پیشینه در می‌یابیم که در سال‌های اخیر سعی دانشمندان بر تولید الگوریتم‌هایی است که اعداد تصادفی را با سرعت و پراکندگی بیشتری تولید کنند. در این مقاله به بررسی روشی کارا در تولید اعداد تصادفی با کمک الگوریتم میانضربی و تابع آشوبگونه لوجستیک می‌پردازیم که تعداد کافی از اعداد تصادفی را با سرعت مناسب و پراکندگی بالا تولید می‌کند. قبل از توضیح سیستم پیشنهادی ابتدا به کاربرد اعداد تصادفی در رمزنگاری می‌پردازیم.

۲-کاربرد اعداد تصادفی در رمزنگاری!

با توجه به کاربرد روزافزون کامپیوتر، حفظ امنیت و تایید صحت اطلاعات، روز به روز اهمیت بیشتری پیدا می‌کند. اطلاعت نظامی، دولتی و حتی پزشکی قبل از مخابره در شبکه باید در قالب‌های امن قرار بگیرند تا از دسترسی بدون اجازه دیگران در امان باشند. این قالب‌های امن از طریق الگوریتم‌های رمزنگاری فراهم می‌شود که از متداول‌ترین این روش‌ها می‌توان به رمزنگاری براساس RSA و DES اشاره کرد. عملکرد مناسب تابع رمز در این روش‌ها ارتباط مستقیمی با کیفیت تولید کلید رمز دارد که اساس تولید آن بر پایه اعداد تصادفی است. تولید دنباله های بسیار بلند در الگوریتم کلید رمز در شیوه RSA نیاز مبرم این روش را به داشتن مولدی تصادفی که داده های تولید اش از هر نظر برای کار در یک الگوریتم رمزنگاری مناسب باشد را محیا می کند در حقیقت، هر قدر اعداد تصادفی تولید شده پراکنده‌تر، غیرقابل پیش‌بینی‌تر و از سرعت بالاتری در تولید برخوردار باشند، کلید رمز نیز از کیفیت بالاتری برخوردار است. به عبارت دیگر احتمال تشخیص کلید رمز از روی داده‌های قبلی کمتر است [۱۷].

در رمزنگاری تصاویر، با توجه به حجم زیاد داده‌های تصویری و ویدیویی و نیز بلندی طول کلید رمز استفاده از روش‌های مناسب در تولید اعداد تصادفی ضروری به نظر می‌رسد. از این رو ایجاد بهبود در روش‌های تولید اعداد تصادفی کمک شایانی به الگوریتم‌های رمزنگاری می‌نماید [۱۷].

با استفاده از توابع آشوب در رمزنگاری امکان تولید کلیدهایی با طول بسیار بلند همراه با الگوریتمی ساده، سریع و ایمن فراهم شده است، همچنین با توجه به فضای بزرگ کلید در توابع آشوب این روش در برابر حملاتی چون Brute force نیز بسیار مقاوم است [14]. در انتها باید گفت که علاوه بر حملات عمدی این الگوریتم نسبت به تغییراتی بسیار کوچک در کلید بسیار حساس بوده، حتی با در دست داشتن مقادیر تقریبی کلید امکان شکستن رمز برای حمله کننده وجود ندارد.

در بخش سوم به معرفی الگوریتم میانضربی و ضعف‌های این الگوریتم در تولید تعداد مناسب اعداد تصادفی می‌پردازیم.

۳-الگوریتم شبه تصادفی به روش میانضربی!

در این روش ابتدا دو عدد که تعداد ارقام مساوی دارند انتخاب می‌شود، (مانند a و b) سپس حاصل ضرب این دو عدد محاسبه

[3] Chaotic Piecewise Linear

و n رقم میانی آن انتخاب، (مقدار $result_1$) و بر 10^n تقسیم می‌شود تا عددی بین صفر و یک حاصل شود (عدد شماره یک- num_1)، این بار مقدار $result_1$ در عدد b ضرب و دوباره n رقم میانی آن انتخاب، (مقدار $result_2$) و این عدد نیز بر 10^n تقسیم می‌شود (عدد شماره دو- num_2) این بار مقدار $result_1$ در مقدار $result_2$ ضرب می‌شود و مراحل بالا به همین صورت ادامه می‌یابد. در رابطه (۱) الگوریتم کار به شکل زیر نشان داده شده است.

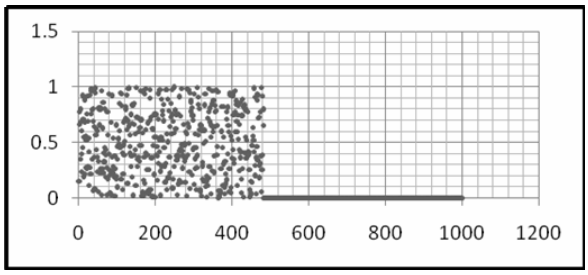
$$c = a \times b \Rightarrow d = c \bmod 10^{\frac{n}{2}} \qquad (1)$$

$$e = (c-d)/10^{\frac{n}{2}} \Rightarrow result_t = e \bmod 10^n$$

$$num_t = result/10^n$$

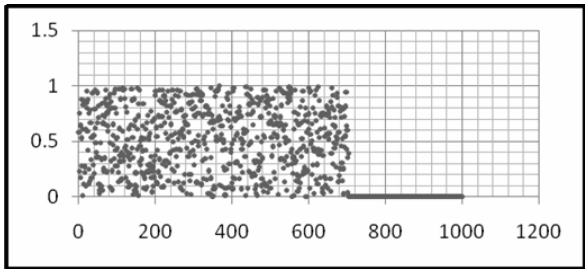
برای تست این الگوریتم چند سری را به صورت تصادفی بررسی کرده و مشکلات هر یک را بیان می‌کنیم.

سری اول که با اعداد هسته (۶۵۲۲,۷۲۲۹) شروع می‌شود. مطابق شکل ۷ تعداد ۴۸۲ عدد تصادفی تولید شده ولی بعد از رسیدن به عدد صفر این عدد تا پایان تکرار می‌شود.



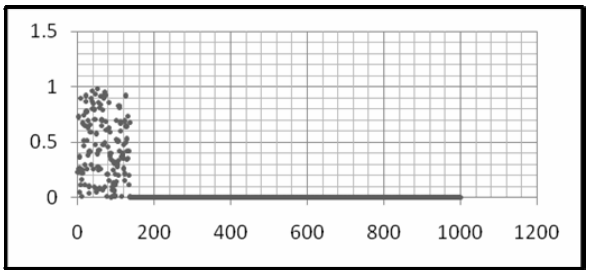
شکل ۷- سری اول تولید کننده میانضربی

سری دوم، که با اعداد هسته (۹۷۱۲,۶۵۴۷) شروع می‌شود. مطابق شکل ۸ تعداد ۷۱۵ عدد تصادفی تولید شده ولی بعد از رسیدن به عدد صفر این عدد تا پایان تکرار می‌شود.



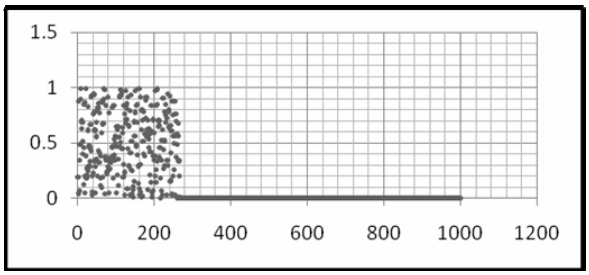
شکل ۸- سری دوم تولید کننده میانضربی

سری سوم که با اعداد هسته (۹۶۳۲,۱۴۷۸) شروع می‌شود. مطابق شکل ۹ تعداد ۱۴۳ عدد تصادفی تولید شده ولی بعد از رسیدن به عدد صفر، این عدد تا پایان تکرار می‌شود.



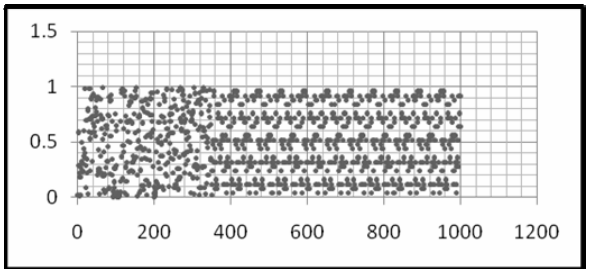
شکل ۹- سری سوم تولید کننده میانضربی

سری چهارم که با اعداد هسته (۲۱۶۳,۸۴۱۰) شروع می‌شود. مطابق شکل ۱۰ تعداد ۲۷۱ عدد تصادفی تولید شده ولی بعد از رسیدن به عدد صفر این عدد تا پایان تکرار می‌شود.



شکل ۱۰- سری چهارم تولید کننده میانضربی

سری پنجم که با اعداد هسته (۲۰۳۶,۶۳۹۵) شروع می‌شود. مطابق شکل ۱۱ تعداد ۱۰۰۰ عدد تصادفی تولید شده و به عدد صفر برخورد نکرده ولی به حلقه تکرار وارد شده است.



شکل ۱۱- سری پنجم تولید کننده میانضربی

با توجه به تعداد کم اعداد تصادفی ایجاد شده در این روش در می‌یابیم که استفاده از آن در رمزنگاری منجر به تولید کلیدهای نامناسب و قابل پیش‌بینی می‌شود. در حقیقت می‌توان اثبات کرد با شروع از هسته‌های تصادفی در روش میانضربی بزرگ‌ترین مشکل برخورد به صفر است که باعث به وجود آمدن حلقه صفر می‌شود.

در بخش بعد به معرفی پدیده آشوب می‌پردازیم و ویژگی‌های منحصر بفرد آن را در تابع آشوبگونه لوجستیک بررسی می‌کنیم.

۴-آشوب و نگاشت آشوبگونه لوجستیک!

آشوب به هر زبانی که ترجمه شود معنی آن دلالت بر رفتاری تصادفی و غیر قابل پیش‌بینی دارد. در حقیقت بار معنایی این لغت

در بر گیرنده بهم ریختگی ناخواسته و اغتشاش است. با این حال در دنیای دانشمندان، این رفتار غیرقابل پیش‌بینی به هیچ وجه ناخواسته و نامطلوب نیست.

امروزه نیز رفتار دینامیک آشوبگونه توجه فراوانی را در بسیاری از زمینه‌های تحقیقاتی به خود معطوف ساخته است که این امر سبب پیشرفت قابل ملاحظه‌ای در مطالعه آشوب شده است.

رابطه تابع لوجستیک که در معادله (۲) نشان داده شده است یک مدل از سیستم‌های دینامیک غیرخطی است که اغلب برای نشان دادن رشد بیولوژیکی جمعیت به کار می‌رود. در سال ۱۹۷۶ مشخص شد که این مدل به ظاهر ساده دارای سیستم دینامیکی پیچیده‌ای است. امرزوه الگوریتم لوجستیک در زمینه‌های مختلفی از نظریه آشوب مانند آشوب در رمزنگاری، رشد بسیار زیادی داشته است. تابع آشوبگونه لوجستیک در حالت گسسته به صورت معادله (۲) می‌باشد [۱۵، ۱۶]:

$$x_{n+1} = f(x_n) = r.x_n(1-x_n) \qquad (2)$$

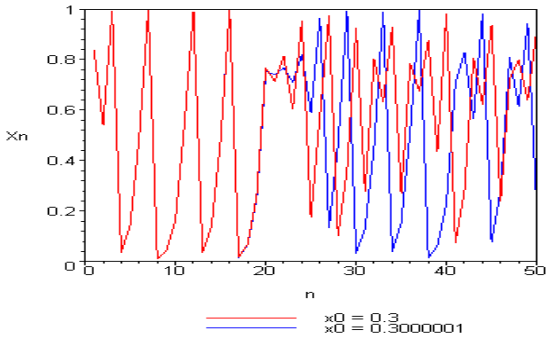
که $x_n \in [0..1]$ پارامتری برای اندازه‌گیری جمعیت در تولید n ام، x_0 جمعیت اولیه، r نرخ رشد و عددی ثابت است. r بین صفر و چهار محدود می‌شود تا فاصله صفر تا یک روی خودش قرار گیرد [۱۵، ۱۶].

در ادامه به دو مورد از ویژگی های بسیار مهم تابع آشوبگونه لوجستیک اشاره می‌کنیم.

۴-۱-حساسیت به شرایط اولیه

یک بستر جذب، مجموعه ای از مقادیر در فضای حالت است که سیستم در طول زمان و یا در اثر تکرار به آنها وارد می‌شود. یک بستر جذب می‌تواند یک نقطه ثابت، یک مجموعه از نقاط که بطور منظم ملاقات می‌شوند، یک حلقه، یک مسیر پیچیده، و یا تعدادی نامتناهی از نقاط باشد. یک بستر جذب N نقطه‌ای که در آن $N = \infty$ است را بستر جذب نامانوس می‌نامند که این نوع بستر جذب مختص سیستم‌های آشوبگونه است. بر این اساس، تعریف سیستم‌های آشوبگونه عبارتست از سیستم‌هایی که دارای تعداد نامحدودی بستر جذب بوده و به شرایط اولیه حساس هستند. حساسیت به شرایط اولیه در یک سیستم دینامیک، بر این اساس است که اختلاف بسیار کوچک در مقادیر اولیه، سبب بوجود آمدن نتایج بسیار متفاوت گردد. این حساسیت توسط کمیتی به نام نمای لیاپانوف اندازه‌گیری می‌شود. این عدد ضریبی از زمان و یا مقدار یک نما است که منعکس کننده نرخ انحراف از مسیر دینامیکی سیستم است. مسیر سیستم، خط سیری از رفتار سیستم در فضای حالت است که از نقاط اولیه سیستم شروع و به نقاط جذب منتهی می‌شوند [5].

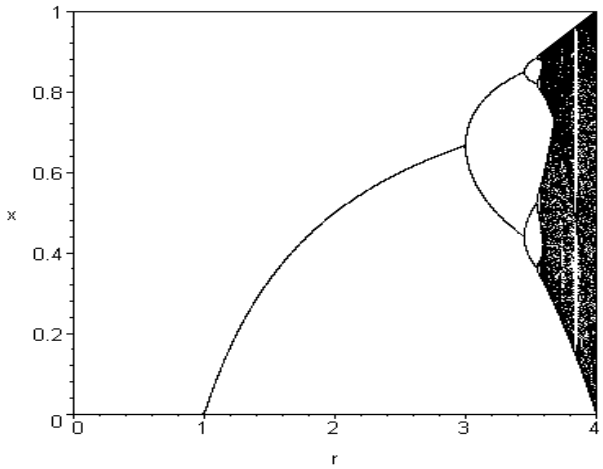
همانطور که در شکل ۱۲ مشخص است، با تغییر کوچکی در مقدار های اولیه در تابع آشوب بعد از گذشت چند تکرار شرایط کاملا متفاوتی را خواهیم داشت و این مساله به خاطر وجود بستر های جذب نا متنهاهی در پارامتری خاص از نگاشت آشوب است. به عنوان مثال، اگر داده‌های اولیه ۰/۳ و ۰/۳۰۰۰۰۰۱انتخاب شود، سیستم آشوبگونه لوجستیک بعد از ۲۵ تکرار رفتار متفاوتی از خود نشان می‌دهد. که به وضوح در شکل ۱۲ مشخص است.



شکل ۱۲- حساسیت زیاد نگاشت آشوب به شرایط اولیه

۴-۲-دوشاخه‌ای شدن

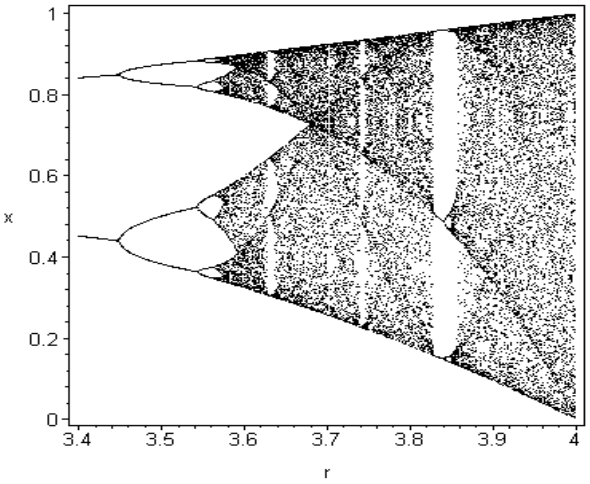
یکی از ویژگی‌های بسیار مهم نگاست لوجستیک پدیده دو شاخه‌ای شدن است که با افزایش r در فضای حالت رخ می‌دهد. در واقع دو شاخه شدن سبب دو برابر شدن پریود می‌شود. یعنی باعث می‌شود که بستر جذب N نقطه‌ای به بستر جذب $2N$ نقطه‌ای تبدیل گردد. این امر زمانی رخ می‌دهد که پارامتر دو شاخه‌ای شدن r تغییر کند. برای مقادیر بزرگتر r، ترتیب x_n ها به یک مقدار ثابت و یا مدار متناوب منتهی نمی‌شود بلکه یک رفتار آشوبگونه دارد [۵].



شکل ۱۳- نمودار دوشاخه‌ای شدن تابع لوجستیک در $0 < r < 4$

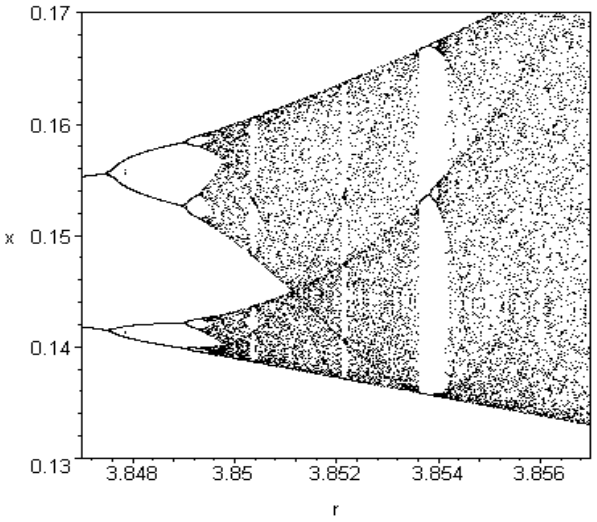
در شکل ۱۳ نمودار x بر حسب r رسم شده است. طریقه رسم این نمودار بدین ترتیب است که ابتدا برای r، مقدار شروع بازه را در نظر می‌گیریم و سپس یک مقدار اولیه دلخواه برای x انتخاب کرده و مقادیر بعدی x را به ترتیب و با توجه به نگاشت مورد نظر بدست می‌آوریم تا رفتار طولانی مدت سیستم بدست آید و رفتار گذرا حذف شود یعنی از ۳۰۰ مقدار اول صرفنظر می‌کنیم و نمودار را برای $300 < n < 350$ رسم می‌کنیم. همه نقاط بدست آمده برای x را روی همان مقدار r درج می‌کنیم. حال با توجه به گام در نظر گرفته شده مقدار r را افزایش داده و همین عمل را تکرار می‌کنیم. اعمال فوق را تا انتهای بازه در نظر گرفته شده برای r، تکرار می‌کنیم.

در شکل ۱۴ نمودار برای $3.4 < r < 4$ با بزرگنمایی بیشتر رسم شده است. لازم بذکر است که سیستم برای همه مقادیر $r > 3.57$ آشوبگونه نیست. به عبارت دیگر در بازه $3.57 < r < 4$، مخلوطی از نظم و آشوب دیده می‌شود. بطوریکه یک تغییر کوچک در r، می‌تواند سیستم را پایدار کرده و یا بر عکس در وضعیت آشوبگونه قرار دهد. در این حالت گفته می‌شود که پنجره‌هایی از نظم در میان آشوب بوجود آمده‌اند. همانطور که در شکل ۱۳ مشخص است، دیاگرام دو شاخه شدن ابتدا پریودهای ۲، ۴، ۸، ۱۶ و ... تولید می‌کند. سپس رفتار آشوبگونه آغاز می‌شود که سیستم در این حالت هیچ پریود منظمی ندارد. اما زمانیکه روند حرکتی سیستم پیچیده‌تر می‌شود، پنجره‌هایی در اثر پریودهای فرد در دیاگرام دو شاخه‌ای پدید می‌آید [5].



شکل ۱۴ - نمودار دو شاخه شدن تابع لوجستیک در $3.4 < r < 4$

در شکل ۱۵ برای حالت $3.847 < r < 3.857$ نیز نمودار با مقیاس بزرگتر رسم شده است. بعلاوه ملاحظه می‌شود که این قسمت بزرگ شده از نمودار، شبیه نمودار اصلی است که به این پدیده ویژگی خودتشابهی می‌گویند که یکی از ویژگی‌های بسترهای جذب نامانوس می‌باشد.



شکل ۱۵ -دو شاخه شدن تابع لوجستیک در $3.847 < r < 3.857$

۵-ارایه الگوریتم پیشنهادی میانضربی لوجستیک!

همانطور که مشاهده شد، بزرگترین مشکل در تولید دنباله‌های میانضربی رسیدن به مقدار صفر و گیرکردن در حلقه تکرار صفر است. همچنین در بعضی موارد می‌توان شاهد بوجود آمدن حلقه‌ای به طول مشخص در دنباله بود که با شروع از بعضی مقادیر به عنوان هسته این اتفاق می‌افتد. حال اگر مقادیر مناسبی برای هسته انتخاب شود، می‌توان موارد برخورد به صفر را با کمک تابع لوجستیک اصلاح کرد. بدین صورت که ابتدا الگوریتم میانضربی داده‌های تصادفی خود را تولید می‌کند، هنگامی که مقدار تولیدی صفر شد، تابع آشوبگونه لوجستیک با مقدار اولیه $x_0 = 0.6$ در $r = 4$ داده‌ای آشوبگونه را تولید و به عنوان هسته جدید در اختیار تابع میانضربی قرار می‌دهد، تا این الگوریتم روند تصادفی خود را حفظ کرده و در حلقه صفر نماند. شبه کد الگوریتم در رابطه (۳) آمده است. در ادامه به بررسی چند دنباله شبه تصادفی که به کمک سامانه میانضربی لوجستیک تولید شده‌اند، می‌پردازیم.

$$c = a \times b \Rightarrow d = c \bmod 10^{\frac{n}{2}} \qquad (3)$$

$$e = (c - d)/10^{\frac{n}{2}} \Rightarrow result_t = e \bmod 10^n$$

$$num_t = result/10^n$$

$$if\ (num_t == 0) then$$

{

$$x_{n+1} = x_n r(1 - x_n)$$

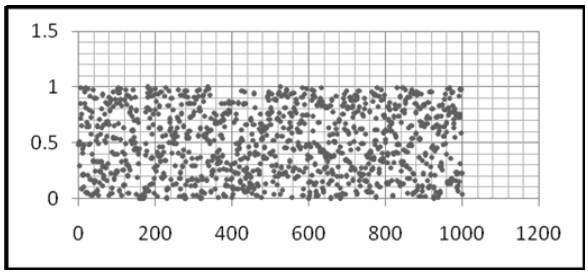
$$b = x_{n+1}$$

$$x_n = x_{n+1}$$

}

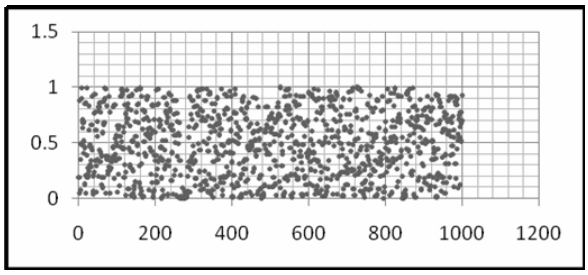
سری اول با هسته‌های (۹۰۸۱,۳۰۲۶) شروع می‌شود. مطابق شکل ۱۶ این سری دو بار در شماره‌های ۱۶۳ و ۱۶۹ به مقدار

صفر رسیده که هر بار با اصلاح مقدار هسته، روند تصادفی خود را دنبال و ۱۰۰۰ عدد تصادفی تولید کرده است.



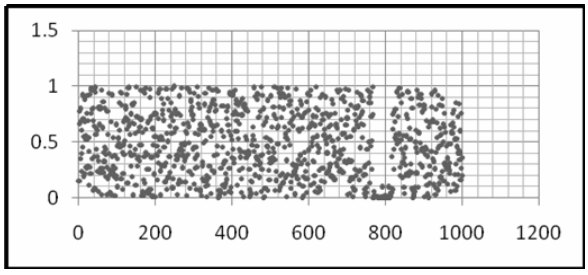
شکل ۱۶- سری اول دنباله میانضربی لوجستیک

سری دوم با هسته‌های (۸۴۱۰,۲۱۶۳) شروع می‌شود. مطابق شکل ۱۷ این سری دو بار در شماره‌های ۲۷۱ و ۲۷۶ به مقدار صفر رسیده که هر بار با اصلاح مقدار هسته، روند تصادفی خود را دنبال و ۱۰۰۰ عدد تصادفی تولید کرده است.



شکل ۱۷- سری دوم دنباله میانضربی لوجستیک

سری سوم با هسته‌های (۷۲۲۹,۶۵۲۲) شروع می‌شود. مطابق شکل ۱۸ این سری دوازده بار در شماره‌های ۴۸۳، ۷۸۴، ۷۸۵، ۷۸۶، ۷۸۷، ۷۸۹، ۷۹۲، ۷۹۷، ۸۰۱، ۸۰۴، ۸۰۵، ۸۰۹ به مقدار صفر رسیده که هر بار با اصلاح مقدار هسته، روند تصادفی خود را دنبال و در نهایت ۱۰۰۰ عدد تصادفی تولید کرده است.



شکل ۱۸- سری سوم دنباله میانضربی لوجستیک

دیده می‌شود، حتی در دنباله آخر که با انتخاب هسته‌های نامناسب منجر به تولید دوازده مقدار صفر شده، همچنان دنباله به روند تصادفی خود ادامه داده و با سرعت مناسبی اعداد تصادفی را تولید کرده است. در ادامه بعد از معرفی تست مونت کارلو به تست روش پیشنهادی و مقایسه آن با نتایج حاصل از روش میانضربی پرداخته شده است.

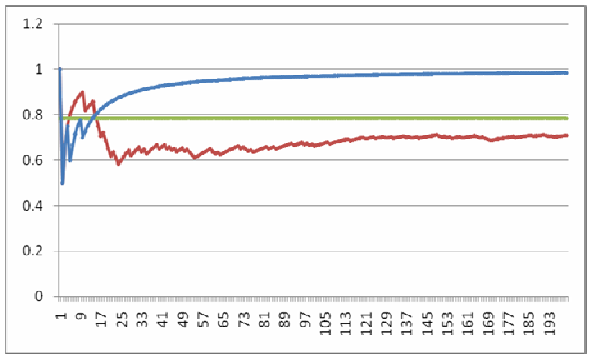
۶-تست مونت کارلو!

در روش تست مونت کارلو دایره‌ای به مساحت یک که مرکز آن منطبق با مرکز دستگاه مختصات دکارتی است در نظر گرفته می‌شود. در نتیجه مساحت بخشی از آن که در ربع اول محورهای مختصات واقع است از رابطه (۴) محاسبه می‌شود [۱۷]:

$$S = \pi.r^2 / 4 = (3.14*1)/4 = 0.785 \qquad \pi = 3.14 \qquad (4)$$

حال توسط روش میانضربی لوجستیک اعداد تصادفی تولید و خروجی‌ها دو به دو به عنوان نقاطی تصادفی در دستگاه مختصات دکارتی در نظر گرفته می‌شود و با توجه به تعداد نقاط داخل دایره (m) و کل نقاط موجود در صفحه (n) بر اساس رابطه زیر مساحت دایره تخمین زده می‌شود. اگر P مساحت تخمینی دایره باشد داریم $P = m/n$. بر این اساس هر اندازه روش زودتر به جواب دقیق مساحت برسد و همگراتر باشد بهتر است.

با توجه به توضیحات بالا و تفسیر نتایج تست مونت کارلو در نرم‌افزار اکسل نمودار شکل ۱۹ بدست آمده است که نشان از بهبود کارایی ایجاد شده در روش میانضربی لوجستیک بعلت استفاده از نگاشت آشوبگونه لوجستیک دارد.



شکل ۱۹- نتایج مقایسه روش میانضربی با روش پیشنهادی نمودار آبی: روش میانضربی، نمودار سبز: مساحت ربع دایره، نمودار قرمز: روش پیشنهادی

۷-نتیجه‌گیری!

اعداد تصادفی در رمزنگاری نقش بسزایی دارند. بنابراین نیاز به الگوریتمی می‌باشد که اعداد تصادفی را با کارایی بالاتر و مناسب جهت تولید کلید رمز در الگوریتم‌های رمزنگاری تولید کند. در این مقاله با توجه به مقایسه الگوریتم پیشنهادی با الگوریتم میانضربی مشاهده شد، اعداد تصادفی تولید شده توسط الگوریتم پیشنهادی طبق تست مونت‌کارلو دارای پراکندگی بیشتری است و استفاده از این الگوریتم باعث می‌شود در الگوریتم‌های رمزنگاری، کلید رمز با کیفیت بالاتری تولید شود که به سادگی قابل تشخیص به وسیله حمله کنندگان نباشد. در پایان باید اشاره کرد که، با ورود تابع آشوبگونه لوجستیک در الگوریتم میانضربی پیچیدگی الگوریتم بالا رفت و به همین نسبت سرعت تولید اعداد تصادفی نیز کاهش یافت ولی باعث شد در هر شرایطی تعداد مناسب اعداد تصادفی،

توسط مولد تولید شود. بنابراین برای مطالعات بعدی پیشنهاد می‌شود سایر توابع آشوب مورد بررسی قرار گیرد و بهترین آنها از نظر پیچیدگی زمانی و بالاترین پراکندگی در ترکیب با مولدهای تصادفی کلاسیک قرار گیرد تا نهایتاً به بهترین عملکرد ترکیب توابع کلاسیک و آشوب در تولید اعداد تصادفی برسیم.

۸-سپاسگزاری

این پژوهش با استفاده از اعتبارات پژوهشی پژوهشکده دانشجویی دانشگاه صنعتی شاهرود انجام گردیده است.

مراجع

[۱۵] رحیم‌اف، حامد، جاهدمطلق، محمدرضا، مزینی، ناصر، **طراحی و پیاده‌سازی حافظه انجمنی با استفاده از شبکه عصبی آشوبگونه**، پایان‌نامه کارشناسی ارشد، دانشگاه علم‌و صنعت ایران، ۱۳۸۵.

[۱۶] فرهادی، محسن، جاهدمطلق، محمدرضا، مزینی، ناصر، **استفاده از تئوری آشوب در سیستم‌های بهینه‌سازی فازی**، پایان نامه کارشناسی ارشد، دانشگاه علم‌وصنعت ایران، ۱۳۸۷.

[۱۷] رشیدی، رحیم، محمد علی، **ارایه الگوریتمی برای تولید اعداد شبه تصادفی با کارایی بالا و استفاده از آن در مدل سازی و ارزیابی سیستم های کامپیوتری**، همایش ملی مهندسی کامپیوتر، برق و فناوری اطلاعات، دوم،۳۳۲-۳۳۵، همدان، ۱۳۸۸.

[1] R. Jain, “Art of Computer System Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling”, Wiley Computer Publishing, John Wiley & Sons, Inc., 2005.

[2] V. V. Kolesov, R. V. Belyaev and G. M. Voronov, “A digital random-number generator based on the chaotic signal algorithm”, journal of communications technology and electronics, vol. 46, pp. 1258-1263, 2001.

[3] M. Babaee, H. Rahimov, M. Farhadi, M.R. Jahed Motlagh, “Random sequence generator based on LFSR”, International Conf. on Intelligent systems and technologies, Paris, France, 2010.

[4] C.J. Kung, and H.C. Tang, “Criterion of Spectral Test for Linear Congruential Random Number Generators”, Journal of Science and Engineering, Vol. 12, No. 3, pp. 365-369, 2009.

[5] T. Stojanovski and L. Kocarev, “Chaos-based random number generators – part 1: practical realization”, IEEE Transactions on circuits and systems I: Fundamental Theory and applications, vol. 48, pp. 281-288, 2001.

[6] T. Stojanovski, J. Pihl, and L. Kocarev,“Chaos-based random number generators – part 2: Practical realization”, IEEE Transactions on circuits and systems I: Fundamental Theory and applications, vol. 48, pp. 382-385, 2001.

[7] L. Kocarev and G. Jakimoski, “Pseudorandom bits generated by chaotic maps” IEEE Transactions on circuits and systems I: Fundamental Theory and applications, vol. 50, pp. 123-126, 2003.

[8] S.M. Fu, Z. –Y. Chen, and Y.A. Zhou, “Chaos-based random number generators”, Computer research and development, vol. 41, pp. 749-754, 2004.

[9] J. Liu, “Design of chaotic random sequence and its application”, Computer Engineering, vol. 31, pp. 150-152, 2005.

[10] Y. Wang, H. Shen, and X. Yan, “Design of a chaotic random number generator”, Chinese Journal of Semiconductors, vol. 26, pp. 2433-2439, 2005.

[11] L. Wang, F.P. Wang, and Z.J. Wang, “Novel chaos based pseudorandom number generator”, Acta Physical Sinica, vol. 55, pp. 3964-3968, 2006.

[12] S. Ergun, and S. Ozoguz, “Truly random number generators based on a non-autonomous chaotic oscillator”, AEU-International J. Electronics & Communications, vol. 62, pp. 235-242, 2007.

[13] Y. Hu, X. Liao, K.W. Wong, and Q. Zhou, “A true random number generator based on mouse movement and chaotic cryptography”, Chaos solitons and fractals, vol. 40, pp. 2286-2293, 2009.

[14] L. Shujun, and Z. Xuan, ‘‘On the Security of an Image Encryption Method’’, IEEE International Conference on Image Processing, vol. 2, pp. 925-928, 2002