TEEGES

Technovations of Electrical Engineering in Green Energy System

Review Article (2023) 2(3):94-129

An Overview of Common Post-Processing Methods in Random Number Generators with an Emphasis on Use in Renewable Systems

Mohsen Mousavi[1], *PhD*.

[1]*Faculty of Applied Sciences, Malek-Ashtar University of Technology, Isfahan, Iran*

Abstract:

One of the main components in the security of cryptography systems is random numbers. Random numbers are often used in the generation of secure keys, digital signatures, and other cryptographic systems. Random numbers generated by an algorithm are called pseudo-random numbers. Although pseudo-random numbers have good statistical properties, they have the problem of periodicity. For this reason, to generate random numbers, the true random number generator method is used, which uses a physical entropy source to generate random numbers. In the real random number generator method, due to the instability of electric circuits, a post-processing step is needed so that the generated numbers have acceptable statistical characteristics. In this paper, an overview of the common post-processing methods of random number generators has been done, so that some of the introduced methods are also used in the generation of quantum random numbers. An important point is the role of renewable energies in the design and construction of devices with limited computing power. For this purpose, in the final part of this article, a fast and optimal post-processing method is introduced in terms of hardware implementation, which can be used in mobile phones, smart cards and devices that have limited computing power.

Keywords: Random number generator, Post-processing methods, Cryptography.

Received: 09 April 2023
Revised: 28 April 2023
Accepted: 12 July 2023
Corresponding Author: Dr. Mohsen Mousavi, m.mousavi@mut-es.ac.ir
DOI: 10.30486/teeges.2023.1984909.1069

مروری بر روش‌های پسپردازش متداول در مولدهای اعداد تصادفی با تاکید بر استفاده در سامانه‌های تجدید پذیر

فناوری‌های نوین مهندسی برق در سیستم انرژی سبز

TEEGES

مقاله مروری

مروری بر روش‌های پساپردازش متداول در مولدهای اعداد تصادفی با تاکید بر استفاده در سامانه‌های تجدید پذیر

سید محسن موسوی[1]، دکتری.

۱- مجتمع علوم کاربردی، دانشگاه صنعتی مالک اشتر، اصفهان، ایران

چکیده:

یکی از مؤلفه‌های اصلی در امنیت سامانه‌های رمزنگاری، اعداد تصادفی است. اعداد تصادفی، اغلب در تولید کلیدهای امن، امضاهای دیجیتال و سایر سامانه‌های رمزنگاری، مورداستفاده قرار می‌گیرند. اعداد تصادفی تولیدشده توسط یک الگوریتم را اعداد شبه تصادفی می‌نامند. اعداد شبه تصادفی اگرچه که دارای ویژگی‌های آماری خوبی هستند اما مشکل تناوبی بودن را دارند. به همین علت، برای تولید اعداد تصادفی، از روش مولد اعداد تصادفی واقعی استفاده می‌گردد که از یک منبع آنتروپی فیزیکی برای تولید اعداد تصادفی استفاده می‌کند. در روش مولد اعداد تصادفی واقعی، به دلیل بی‌ثباتی مدارهای الکتریکی یک مرحله پساپردازش نیاز بوده تا اعداد تولیدشده دارای ویژگی‌های آماری قابل‌پذیرشی باشند. در این مقاله، یک مرور کلی بر روی انواع روش‌های متداول پساپردازش مولدهای اعداد تصادفی، انجام‌گرفته است به‌طوری‌که برخی از روش‌های معرفی‌شده، در تولید اعداد تصادفی کوانتومی نیز کاربرد دارند. نکته مهم، نقش انرژی‌های تجدید پذیر در طراحی و ساخت دستگاه‌های با توان محدود محاسباتی است. به همین منظور در بخش پایانی این مقاله، یک روش پساپردازش سریع و بهینه از نظر پیاده‌سازی سخت‌افزاری معرفی می‌گردد که مولد معرفی‌شده را می‌توان در تلفن‌های همراه، کارت‌های هوشمند و وسایلی که دارای توان محاسباتی محدود هستند استفاده نمود.

واژه‌های کلیدی: مولد اعداد تصادفی، روش‌های پساپردازش، رمزنگاری

تاریخ ارسال مقاله: ۱۴۰۲/۰۱/۲۰

تاریخ بازنگری مقاله: ۱۴۰۲/۰۲/۰۸

تاریخ پذیرش مقاله: ۱۴۰۲/۰۴/۲۱

نویسنده‌ی مسئول: دکتر سید محسن موسوی، m.mousavi@mut-es.ac.ir

DOI: 10.30486/teeges.2023.1984909.1069

۱-مقدمه

مولدهای اعداد تصادفی استاندارد برای تولید رشته‌های تصادفی با توزیع یکنواخت[۱] طراحی‌شده‌اند. مرحله پساپردازش[۲] موجود در این مولدها، دنباله بیتی خام[۳] را به رشته‌ای از بیت‌های تصادفی تبدیل نموده که توزیع حاصل از آن‌ها تا حد ممکن به توزیع یکنواخت نزدیک است [۱]. بنابراین، هدف اصلی مرحله پساپردازش، تولید اعداد تصادفی است. اغلب مولدهای اعداد تصادفی[۴] از خانواده‌ای از مولدهای تصادفی استفاده می‌کنند که در آن‌ها سعی می‌شود اریب (یا بایاس) و همبستگی[۵] به وجود آمده در رشته‌ها به علت نقص[۶] موجود در دستگاه تولید اعداد تصادفی، برطرف شود. این اریب و همبستگی حتی در دستگاه‌های تولید اعداد تصادفی که از منابع با آنتروپی بالا[۷] استفاده می‌کنند نیز می‌تواند وجود داشته باشد.

برخی سخت‌افزارهای مولدهای اعداد تصادفی با جمع دودویی خروجی‌های چندین منبع مختلف یا با بازخورد کردن خروجی آن‌ها به یک تابع درهم‌ساز رمزنگاری[۸]، سعی می‌کنند از چندین منبع مختلف برای تولید اعداد تصادفی استفاده کنند [۲]. فون نویمان[۹] یک روش ساده‌ای برای تولید اعداد تصادفی غیر اریب[۱۰] پیشنهاد داده که در آن زوج بیت‌های 00 و 11 از رشته بیت اصلی حذف‌شده و زوج‌های 01 و 10 به ترتیب با بیت‌های 0 و 1 جایگزین می‌شوند [۳]. اگر اریب منظمی در رشته بیتی وجود داشته باشد در این صورت با استفاده از روش فون نویمان می‌توان این اریب را از بین برد. البته با این روش حداقل نصف بیت‌های رشته ورودی حذف خواهند شد و نرخ بیتی به $\frac{1}{4}$ کاهش پیدا خواهد نمود [۴-۵].

تولید یک دنباله از اعداد تصادفی با آنتروپی بالا، عملکرد صحیح یک مولد اعداد تصادفی را تضمین نمی‌کند. به همین علت روش‌هایی وجود دارند که می‌توانند مشکلات منابع ضعیف[۱۱] تولید اعداد تصادفی را برای استفاده در الگوریتم‌های تصادفی شده[۱۲]، تصحیح کنند اما اغلب پروتکل‌ها نمی‌توانند با اعداد تصادفی تولیدشده توسط این الگوریتم‌ها کار کنند [۶]. در حقیقت، در بسیاری از پروتکل‌های رمزنگاری برای داشتن یک پیاده‌سازی امن از پروتکل‌ها و انجام دادن عملیاتی مانند تعهد پیامی[۱۳]، رمزگذاری، اثبات دانش‌صفر[۱۴] و تسهیم‌راز[۱۵] نیاز داریم که اعداد تصادفی استفاده‌شده در پروتکل‌ها، دارای توزیع یکنواخت باشند [۷].

به تازگی استفاده از انرژی تجدید پذیر در دستگاه‌های با توان محدود محاسباتی یکی از چالش‌های مهم است. درواقع، منظور ما از انرژی تجدید پذیر، انرژی‌ای است که از منابعی تأمین می‌شود که همیشه موجود هستند یا می‌توانند در آینده بازیافت شوند. فنّاوری‌های مختلفی برای بهره‌برداری از انرژی‌های تجدید پذیر وجود دارند که یکی از شاخص‌ترین آن‌ها، پنل‌های خورشیدی است. یکی از کاربردهای این تکنولوژی، استفاده در وسایلی است که دارای توان محاسباتی محدود هستند. به همین دلیل، طراحی سخت‌افزارهایی که مناسب این دستگاه‌ها باشد موردتوجه جدی واقع‌شده است. در بخش پایانی این مقاله، یک مولد اعداد تصادفی معرفی می‌گردد که از نظر سخت‌افزاری مناسب این دسته از وسایل با محدودیت محاسباتی است.

۱-۱-پیش‌نیازها

قبل از اینکه مولد اعداد تصادفی را بخواهیم با جزئیات بیشتر توضیح دهیم، لازم است تعریفی ارائه شود تا با استفاده از آن‌ها، دنباله‌های اعداد تصادفی یکنواخت قابل‌پذیرش[۱۶] را مشخص کنیم. یک مفهوم مناسب برای این کار، استفاده کردن از فاصله بین توزیع‌ها است. برای دو توزیع احتمالی X و Y تعریف‌شده بر روی مجموعه پوششی یکسان (منظور از مجموعه پوششی[۱۷]، مجموعه‌ای متشکل از نمادهایی از الفبای متناهی $\mathcal{A}$ است که متغیرها مقادیرشان را از آن اتخاذ می‌کنند)، فاصله آماری بین توزیع‌ها به‌صورت زیر تعریف می‌شود:

$$d(X,Y)=\max_{a\in A}\left|P_X(a)-P_Y(a)\right| \qquad (۱)$$

برای نشان دادن توزیع یکنواخت بر روی $\{0,1\}^d$ از U_d استفاده می‌کنیم. همچنین فرض می‌شود یک RNG ایده‌ال[۱۸]، که توسط یک متغیر تصادفی توصیف می‌شود، از یک توزیع یکنواخت پیروی می‌کند. اصل حداقل آنتروپی به‌طور گسترده برای تعیین کمیت تصادفی بودن یک توزیع احتمال استفاده می‌شود که در ادامه تعریف آن آورده می‌شود.

تعریف ۱: حداقل آنتروپی یک توزیع احتمال مانند X بر روی $\{0,1\}^n$ به‌صورت زیر تعریف می‌شود:

$$H_\infty(X)=-\log_2(\max_{v\in\{0,1\}^n}\Pr ob[X=v]) \qquad (۲)$$

در رمزنگاری، انحراف[۱۹] یک پروتکل عملی از یک پروتکل ایده‌ال با یک پارامتر امنیتی با نماد ε مشخص می‌شود. فاصله آماری معمولاً به‌عنوان یک معیار امنیتی استاندارد استفاده‌شده که در ادامه تعریف می‌شود.

تعریف ۲: دو توزیع احتمال X و Y بر روی دامنه یکسان T را ε-نزدیک گوییم اگر فاصله آماری بین این دو توزیع توسط ε محدودشده باشد:

$$\begin{aligned} \|X-Y\| &\equiv \max_{V\subseteq T}\left|\sum_{v\in V}(\Pr ob[X=v]-\Pr ov[Y=v])\right| \\ &= \frac{1}{2}\sum_{v\in T}|\Pr ob[X=v]-\Pr ob[Y=v]|\le \varepsilon \end{aligned} \qquad (۳)$$

برابری دوم در رابطه (۳) را می‌توان با استقرا گرفتن بر روی اندازه دامنه یا همان $|T|$، به‌صورت زیر اثبات نمود. واضح است که رابطه برای $|T|=1,2$ برقرار است. برای حالت $|T|\ge 3$ همیشه می‌توان v_1 و v_2 را در T پیدا نمود به‌طوری‌که:

$$(\Pr ob[X=v_1]-\Pr ov[Y=v_1])(\Pr ob[X=v_2]-\Pr ov[Y=v_2])\ge 0,$$

آنگاه می‌توان v_1 و v_2 را باهم به‌عنوان یک متغیر جدید v' ترکیب نمود تا دامنه جدید T' تشکیل دهد. حال اگر رابطه برای $S(|T|)$ برقرار باشد آنگاه بحث بالا نشان می‌دهد که رابطه برای $S(|T|+1)$ نیز درست است. ازآنجایی‌که رابطه برای حالت‌های $S(1)$ و $S(2)$ صادق است آنگاه با استفاده از استقرا ثابت می‌شود که $S(|T|)$ برای $T\ge 1$ درست است.

در مولدهای اعداد تصادفی، هدف تولید یک دنباله از اعداد تصادفی است به‌طوری‌که توزیع حاصل از آن تا حد ممکن به توزیع یکنواخت نزدیک باشد. به عبارتی، اگر یک دنباله بیتی خام از طول n در اختیار داشته باشیم و با استفاده از مولد اعداد تصادفی آن را به رشته بیتی از طول m تبدیل کنیم و توزیع حاصل از رشته بیت جدید یک توزیع ϵ-نزدیک به توزیع U_m (توزیع یکنواخت بر روی $\{0,1\}^m$) باشد در این صورت ϵ تا حد ممکن، بر اساس نیازی که داریم، باید مقدار کوچکی باشد.

به‌طورکلی، فاصله آماری، قابلیت تمایز[۲۰] دو توزیع احتمال را توصیف می‌کند. ضریب $1/2$ در رابطه (۳) برای نرمال کردن فاصله آماری استفاده می‌شود تا مقدار آن در فاصله $[0,1]$ بیفتد. درواقع اگر توزیع احتمال X از توزیع احتمال Y دارای خاصیت ε -نزدیک باشد به این معنی است که به‌احتمال‌زیاد X از Y غیرقابل تشخیص بوده و فقط برای احتمال کوچک ε قابل‌تشخیص است. به‌عنوان‌مثال، اگر یک مولد اعداد تصادفی عملی در تعریف ۲ صدق کند آنگاه این مولد ε -نزدیک به مولد اعداد تصادفی ایده‌ال است. توجه شود که پارامترهای امنیتی به‌دست‌آمده از تعریف ۲ قابل‌ترکیب[۲۱] هستند. مفهوم ترکیب‌پذیری[۲۲] برای اولین بار در رمزنگاری کلاسیک برای تحلیل امنیت پروتکل‌های رمزنگاری کلاسیک[۲۳] و به شیوه‌ای پیچیده، مطرح شد و سپس برای رمزنگاری کوانتومی استفاده گردید.

تعریف ۳: یک (k,ε,n,d,m) -استخراج‌کننده یک تابع است که به‌صورت زیر تعریف می‌شود:

$$Ext:\{0,1\}^n\times\{0,1\}^d\to\{0,1\}^m \qquad (۴)$$

به‌طوری‌که برای هر توزیع احتمال X بر روی $\{0,1\}^n$ با خاصیت $H_\infty(X)\ge k$، توزیع احتمال $Ext(X,U_d)$ نسبت به توزیع احتمال یکنواخت بر روی $\{0,1\}^m$ داری ویژگی ε -نزدیک باشد.

به‌طور خلاصه، یک استخراج‌کننده تابعی است که یک مقدار اولیه کوچک شامل d بیت و همچنین n بیت از یک منبع تااندازه‌ای[۲۴] تصادفی را به‌عنوان خروجی می‌گیرد و m بیت رشته تقریباً کامل[۲۵] تصادفی را به‌عنوان خروجی برمی‌گرداند.

تعریف ۴: یک استخراج‌کننده (k,ε,n,d,m)-قوی با نماد $Ext(X,U_d)$ یک استخراج‌کننده‌ای است به‌طوری‌که توزیع احتمال $Ext(X,U_d)$ به توزیع یکنواخت، ε -نزدیک بر روی $\{0,1\}^{m+d}$ است.

توجه داشته باشید مزیت اصلی یک استخراج‌کننده قوی این است که مقدار اولیه ورودی (تصادفی) را می‌توان با افزایش دادن یک پارامتر امنیتی توسط ε، مجدداً استفاده نمود. بنابراین، می‌توان خروجی یک مولد اعداد تصادفی عملی را به بلوک‌های کوچک تقسیم کرد و آن‌ها را توسط یک استخراج‌کننده قوی با همان مقدار اولیه پردازش نمود.

تعریف ۵: خانواده‌ای از توابع درهم‌ساز H که S را به T نگاشت می‌کند را دو-عمومی یا دو-فراگیر[۲۶] گوییم اگر شرط زیر برای همه $x \neq y \in S$ برقرار باشد.

$$P_{h\in H}\{h(x)=h(y)\} \leq \frac{1}{|T|} \quad (۵)$$

در حالت ایده‌آل، هدف طراحی مولدهای اعداد تصادفی است که بیشترین تعداد از بیت‌های خروجی را با کمترین منابع محاسباتی، مانند مدت‌زمان انجام محاسبات یا منابع تصادفی سازی اضافی، تولید کنند. از این نظر، معیارهای تصادفی بودن به‌عنوان راهنمای طراحی مولدهای اعداد تصادفی به کار گرفته خواهند شد. حداقل آنتروپی حاصل از توزیع یک دنباله خام، محدودیتی را برای تعداد بیت‌هایی که می‌توانیم به‌وسیله یک مولد تولید کنیم اعمال می‌کند. اگر رشته بیت‌هایی از طول n، با توزیع X و حداقل آنتروپی $H_\infty(X)=k$، را به‌عنوان دنباله خام در اختیار داشته باشیم می‌توانیم حداکثر k بیت تصادفی را تولید کنیم طوری که توزیع حاصل از آن‌ها به توزیع یکنواخت نزدیک باشد. یک فرآیند تصادفی را (n,k)-منبع می‌نامند اگر خروجی متشکل از n بیت را تولید کند طوری که بیت‌های تولیدشده دارای توزیع X و حداقل آنتروپی $H_\infty(X) \geq k$ باشند.

در بخش‌های بعدی درباره روش‌های مختلف موجود برای تولید دنباله‌های بیتی بحث خواهد شد که توزیع حاصل از خروجی آن‌ها به توزیع یکنواخت نزدیک باشد. همچنین مزیت‌ها و محدودیت‌های روش‌های مختلف تولید اعداد تصادفی بیان می‌گردد. به‌طور دقیق‌تر اینکه در بخش دوم، روش‌های ارزیابی خروجی QRNG شرح داده‌شده و در بخش سوم روش‌های پساپردازش معروف Toeplitz و Trevisan همراه با نحوه پیاده‌سازی آن‌ها آورده شده است. در بخش چهارم، روش پساپردازشی فون‌نویمان برای تولید اعداد تصادفی معرفی‌شده است. در پایان و بخش پنجم، یک پیاده‌سازی از مولد اعداد تصادفی بر اساس ثبات انتقال با پس‌خورد خطی (LFSR) شرح داده می‌شود که نرخ تولید بیت‌های این خروجی برابر با ۴۸۰ مگابیت بر ثانیه بوده که مناسب برای سخت‌افزارهای با توان محدود محاسباتی است.

۲-مولدهای اعداد تصادفی

مولدهای اعداد تصادفی توابعی هستند که منابعی با آنتروپی ضعیف را به مولدهای بیتی با توزیع یکنواخت تبدیل می‌کنند. آن‌ها در ابتدا جهت مطالعه الگوریتم‌های تصادفی شده معرفی شدند ولی در ادامه به یک ابزار پایه‌ای برای استفاده در بخش‌های مختلف علوم کامپیوتر تبدیل شدند. مولدهای اعداد تصادفی و مبحث مرتبط با آن، همانند دیسپرسرها[۲۷]، کندانسورها[۲۸] و گراف‌های بسط دهنده، کاربردهای متعددی دارند و در مبحث مولدهای اعداد شبه تصادفی، کدهای تصحیح خطا[۲۹]، نمونه‌بردارها[۳۰]، گراف‌های بسط دهنده، تقویت‌کننده‌های سخت و غیره مورداستفاده قرار می‌گیرند [۸].

در این بخش، درباره تعدادی از مفاهیم مربوط به مولدها که ارتباط بیشتری با مولدهای اعداد تصادفی کوانتومی (QRNG) دارند بحث خواهیم نمود. انتخاب‌های زیادی برای طراحی مولدهای اعداد تصادفی وجود دارد ولی در انتخاب یک مولد مناسب باید سرعت پیاده‌سازی و نیازهای سخت‌افزاری روش در نظر گرفته شوند. در ادامه تعدادی از مولدهای معروف را بیان می‌کنیم [۱۲-۹].

برای داشتن یک مولد اعداد تصادفی کارآمد جهت تولید حداکثر تعداد ممکن از بیت‌های تصادفی، نیاز داریم که یک تخمین خوبی از آنتروپی در دسترس داشته باشیم و سپس متناسب با آن آنتروپی، یک مولد اعداد تصادفی مناسب طراحی نماییم در غیر این صورت، خروجی تابع مولد، ویژگی‌های مدنظر برای تولید اعداد تصادفی را نخواهد داشت [۱۳].

در ادامه فرض می‌کنیم که یک منبع تصادفی به‌خوبی مشخص‌شده[۳۱] در اختیار داریم. همچنین فرض می‌کنیم که منبع در نظر گرفته‌شده ضوابط و معیارهای آنتروپی ذکرشده در بخش‌های قبل را دارد. فرض می‌کنیم که دنباله خام دارای حداقل آنتروپی شناخته‌شده است. در برخی موارد می‌توان فرض نمود که بیت‌های دنباله خام دارای توزیع‌های مستقل هستند و یا بر اساس فرآیند مارکوف[۳۲] تولید می‌شوند. در بخش‌های بعد، همچنین فرض می‌کنیم که هدف ما دستیابی به یک (n,m,k,ϵ)-مولد است، یعنی تابعی که n بیت حاصل از (n,k)-منبع را به m بیت خروجی تبدیل کند به‌نحوی‌که توزیع حاصل از بیت‌های خروجی یک توزیع ϵ-نزدیک به توزیع یکنواخت باشد و مقدار m تا حد ممکن به k نزدیک شود.

۲-۱-مولدهای قطعی

مولدهای قطعی[۳۳] توابعی از فرم زیر هستند [۲۴]:

$$\mathrm{Ext}:\{۰,۱\}^{n}\rightarrow\{۰,۱\}^{m} \qquad (۶)$$

این توابع رشته بیت‌هایی از طول n را به‌عنوان ورودی می‌گیرند و رشته بیت‌هایی از طول m به‌عنوان خروجی تولید می‌کنند. با توجه به ویژگی قطعی بودن، این توابع موردتوجه قرارگرفته‌اند چون برای تولید خروجی فقط به ورودی تابع وابسته هستند. ولی برخی محدودیت‌ها نیز دارند که باعث می‌شود نتوان آن‌ها را با برخی منابع تصادفی استفاده نمود.

همانند سایر مولدها، در این دسته از مولدها نیز اگر دنباله ورودی مولد، دارای آنتروپی کافی باشد می‌توان خروجی‌ای تولید نمود که توزیع آن نزدیک به توزیع یکنواخت شود. اگر دنباله ورودی یک (n,k)-منبع باشد یک شرط لازم برای اینکه توزیع حاصل از خروجی دنباله به توزیع یکنواخت نزدیک شود این است که $m\leq k$ باشد. متأسفانه، شرط لازم بالا یک شرط کافی نیست چون فقط می‌توان مولدهای قطعی برای برخی ورودی‌ها با توزیع‌های محدود پیدا نمود.

در ادامه با بیان یک استدلال ساده نشان می‌دهیم که طراحی مولدهای قطعی کلی[۳۴]، غیرممکن است. تابعی را از دامنه $\{0,1\}^n$ به برد $\{0,1\}$ را تصور کنید. می‌توان ورودی‌های این تابع را به دو مجموعه تقسیم کرد. مجموعه اول ورودی‌هایی هستند که خروجی یک را تولید می‌کنند (این مجموعه را $\mathrm{Ext}^{-1}(1)$ نامیده) و مجموعه دوم ورودی‌هایی هستند که خروجی صفر را تولید می‌کنند (این مجموعه را $\mathrm{Ext}^{-1}(0)$ می‌نامیم). می‌توان گفت که اندازه یکی از این مجموعه‌ها حداقل برابر با 2^{n-1} است. حال یک ورودی را در نظر بگیرید که دارای توزیع یکنواخت بوده و در مجموعه بزرگ‌تر قرار دارد. این ورودی دارای حداقل آنتروپی برابر با $n-1$ است ولی هر بار مولد را به ازای این ورودی اجرا کنیم خروجی یکسانی تولید خواهد کرد. این موضوع نشان می‌دهد که مولدی وجود ندارد که برای هر ورودی، مناسب و سازگار باشد [۱۴].

البته مولدهایی وجود دارند که برای ورودی‌هایی که متعلق به خانواده مشخصی از فرآیندها که بیانگر منابع منطقی[۳۵] هستند به‌درستی کار می‌کنند. از میان این مولدها، مولدهای قطعی کاربردی[۳۶] برای توزیع‌های قابل نمونه‌برداری نیز وجود دارند که می‌توانند توسط الگوریتم‌های نمونه‌برداری کارآمدی تولید شوند. برای منابع با ویژگی تثبیت-بیت[۳۷] که در آن‌ها دشمن می‌تواند تعدادی از بیت‌های منبع را انتخاب کند مولدهای قطعی کاربردی پیشنهادشده است. به‌تازگی، مولدهای قطعی برای منابع آفینی[۳۸] پیشنهادشده‌اند. همچنین مولدهای قطعی کاربردی برای منابعی که خروجی‌هایی با توزیع غیریکنواخت و تنوع جبری ناشناخته[۳۹] تولید می‌کنند پیشنهادشده است [۱۵-۲۰].

۲-۱-۱-معرفی مقدماتی الگوریتم فون‌نویمان

مولدهای قطعی با طول متغیر، گروه دیگری از مولدهای قطعی را تشکیل می‌دهند که تعاریف آن‌ها اندکی با تعریف ذکرشده در رابطه (۳) متفاوت است. این مولدها در حوزه الگوریتم فون‌نویمان قرار دارند که در قسمت چهارم این مقاله، مفصل در مورد جزئیات آن بحث شده است. الگوریتم فون‌نویمان یک روش قطعی است که بر روی یک توزیع نامشخص می‌توان آن را اعمال نمود و خروجی‌هایی تولید کرد که قبل از تولید، قابل پیش‌بینی نیستند. روش‌های دیگری بر اساس روش فون‌نویمان پیشنهادشده‌اند که در آن‌ها تعداد بیت‌های حذف‌شده دنباله، کاهش‌یافته است. این باعث شده است که کارایی این روش‌ها به کران تئوری اطلاعاتی حاصل از آنتروپی شانون[۴۰] منبع نزدیک باشد. با تصحیح بیشتر روش فون‌نویمان می‌توان به الگوریتم‌هایی دست‌یافت و با استفاده از این الگوریتم‌ها دنباله‌های غیر اریب با شرایط کلی‌تر تولید نمود که ورودی الگوریتم‌ها توسط یک زنجیر مارکوف[۴۱] تولید می‌شود [۲۱-۲۲].

برخی محدودیت‌ها نیز در روش فون‌نویمان وجود دارد. برای مثال، اگر دشمن بتواند اریب ورودی الگوریتم از بیتی به بیت دیگر را تغییر دهد در این صورت روش فون‌نویمان به‌درستی کار نخواهد کرد. به عبارتی، الگوریتم قطعی وجود ندارد که بتوان با استفاده از آن خروجی یکنواخت به ازای ورودی تصادفی $X=(X_1,X_2,\dots,X_n)$ تولید نمود به‌طوری‌که اگر اریب بیت‌های ورودی را به نحوی تغییر دهیم رابطه احتمالاتی زیر برای $0<\delta\leq\frac{1}{2}$ برقرار باشد:

$$\delta\leq\Pr_{x_i}\left(۱\,|x_۱\ x_۲\dots x_{n-۱}=s\right)\leq ۱-\delta \qquad (۷)$$

این نوع از منابع تولید اعداد تصادفی، منابع سانتا-وزیرانی[۴۲] نامیده شده و به‌عنوان مدلی برای منابع تصادفی ضعیف ارائه‌شده‌اند [۲۳]. باوجود محدودیت‌های ذکرشده در بالا، الگوریتم‌های قطعی وجود دارند که می‌توان با استفاده از آن‌ها و منابع سانتا-وزیرانی ضعیف، به شبیه‌سازی الگوریتم‌های تصادفی شده پرداخت. در مقایسه با موارد کاربردی مانند رمزنگاری، سخت‌گیری‌های کمی در شرایط تصادفی سازی دنباله اعداد وجود دارد و منابع ضعیفی که اغلب خروجی‌های یکنواخت تولیدشده توسط آن‌ها در موارد کاربردی قابل‌استفاده نیستند در تصادفی سازی معتبر هستند [۲۵-۲۴].

حتی اگر از یک مولد قطعی استفاده کنیم، استفاده کردن از یک منبع ضعیف برای تولید اعداد تصادفی جهت استفاده در پروتکل‌های تصادفی به‌اندازه کافی، مناسب نیست. اگرچه که اعداد تصادفی ضعیف را می‌توان با طرح‌های امضا[۴۳] (امضاهای دیجیتال) به‌صورت امن استفاده نمود ولی برای رمزگذاری و سایر پروتکل‌های رمزنگاری که نیازمند کلیدهای قوی هستند نمی‌توان استفاده نمود [۲۹-۲۶].

برای کاربردهایی که نیاز به اعداد تصادفی با توزیع نزدیک به توزیع یکنواخت داریم، به‌تازگی یک پیشنهاد ساده مطرح‌شده است. به بیان دقیق‌تر پیشنهادشده است که از دو منبع ضعیف سانتا-وزیرانی مستقل استفاده شود و خروجی آن‌ها با هم ترکیب شوند. با این کار می‌توان به دنباله‌ای از اعداد تصادفی دست پیدا کرد که با استفاده از الگوریتم‌هایی با مدت زمان چندجمله‌ای نمی‌توان آن‌ها را از دنباله‌هایی با توزیع یکنواخت تشخیص داد. با این روش تا زمانی که به یک روش فیزیکی جهت تولید اعداد تصادفی دسترسی داریم، می‌توانیم رشته بیت‌هایی تولید کنیم که توسط الگوریتم‌های کارآمد از رشته بیت‌هایی با توزیع یکنواخت قابل‌تشخیص نیستند. از این روش می‌توان جهت طراحی مولدهای اعداد تصادفی واقعی استفاده نمود که در بیشتر کاربردها مانند رمزنگاری می‌توانند کاربرد داشته باشند.

مولدهای چند منبعی[۴۴] نیز از این مدل برای تولید اعداد تصادفی استفاده می‌کنند. این منابع خروجی، دو یا چند منبع ضعیف را استفاده نموده و با پردازش آن‌ها دنباله‌ای تولید می‌کنند که توزیع آن نزدیک به توزیع یکنواخت است. روش‌های زیادی وجود دارند که به توزیع ورودی‌های به‌هم‌پیوسته[۴۵]، تعداد منابع در دسترس و ویژگی‌های موردنظر دنباله خروجی بستگی دارند.

اخیراً یک مولد ساده برای بلوک‌های n بیتی که از دو منبع ضعیف مستقل تولید می‌شوند پیشنهادشده است. حداقل آنتروپی این منابع برابر با $\frac{n}{2}$ است و از ضرب داخلی بلوک‌های n بیتی، در میدان $GF(2)$، جهت تولید رشته تصادفی در این مولدها استفاده می‌شود. این مسئله باعث می‌شود پیچیدگی محاسباتی تولید رشته بیت تصادفی به محاسبه AND-بیتی[۴۶] دو دنباله کاهش پیدا نماید [۳۱-۳۰].

همچنین روش‌های دیگری برای ترکیب خروجی‌های منابع تصادفی مختلف پیشنهادشده است. در دومین گروه از مولدهای اعداد تصادفی، یعنی مولدهای دارای مقدار اولیه، از ایده ترکیب منابع مختلف استفاده‌شده است. این گروه از مولدها به‌عنوان گروه خاص از مولدهای چند منبعی در نظر گرفته می‌شوند که در آن‌ها یک منبع ضعیف و یک منبع با توزیع یکنواخت کامل وجود دارند. منبع دارای توزیع یکنواخت کامل فقط تعداد کمی بیت تصادفی تولید می‌کند [۳۵-۳۲].

۲-۲-مولدهای دارای مقدار اولیه

همان‌طور که شرح داده شد برای بیشتر توزیع‌های بیتی خام، با کمک اعداد تصادفی اضافی می‌توانیم خروجی‌هایی تولید کنیم که توزیع آن‌ها به توزیع یکنواخت نزدیک شود. مولدهای دارای مقدار اولیه[۴۷] توابعی به شکل زیر هستند.

$$\mathrm{Ext}:\{۰,۱\}^{n}\times\{۰,۱\}^{d}\ \rightarrow\{۰,۱\}^{m} \qquad (۸)$$

این توابع دو ورودی دارند. یکی از ورودی‌ها، دنباله خام از طول n بیت بوده و دیگری یک مقدار اولیه متشکل از d بیت می‌باشد که دارای توزیع یکنواخت است. به ازای ورودی‌های ذکرشده، این مولدها یک رشته بیت از طول m را به‌عنوان خروجی تولید می‌کند. فرض می‌کنیم که d بسیار کوچک‌تر از m است. مقدار اولیه استفاده‌شده در این نوع از مولدها مشابه مقدار اولیه استفاده‌شده در مولدهای اعداد شبه تصادفی بوده و با استفاده از آن می‌توان تضمین نمود مولدهایی وجود دارند که با استفاده از آن‌ها می‌توان خروجی‌هایی تولید کرد که توزیع آن‌ها نزدیک به توزیع یکنواخت است. همچنین طول خروجی‌های تولیدشده توسط این نوع از مولدها تا حد ممکن به بیشترین طول نزدیک می‌شود. یک (k,ϵ)-مولد تابعی است که برای هر ورودی خام از منبعی با حداقل آنتروپی k (k-منبع) دنباله‌ای تولید کند که توزیع آن ϵ-نزدیک به توزیع یکنواخت باشد. مقدار اولیه استفاده‌شده در مولدهای دارای مقدار اولیه، به‌صورت یک کاتالیزور[۴۸] عمل می‌کند و اجازه می‌دهد تا بتوانیم روش‌هایی برای تولید اعداد تصادفی پیدا کنیم که همواره کار کنند.

مولدهای دارای مقدار اولیه برای اولین بار در مبحث الگوریتم‌های تصادفی شده معرفی شدند. با استفاده از روش احتمالاتی، نشان داده‌شده است که همواره مولدهایی وجود دارند که خروجی‌های آن‌ها شامل تقریباً تمام آنتروپی پنهان و قابل‌دسترس ورودی‌های خام تولیدشده توسط یک k-منبع است. برای بلوک‌های ورودی از طول n بیت و تولیدشده توسط یک k-منبع، می‌توان مولدهایی طراحی کرد که اندازه خروجی آن‌ها برابر با $m \approx k + d$ است و توزیع خروجی آن‌ها یک توزیع ϵ-نزدیک به توزیع یکنواخت است و این در حالی است که فقط از یک مقدار اولیه از طول n و مرتبه $\log n$ در پیاده‌سازی آن‌ها استفاده گردد. ساختارهای مختلفی برای این نوع از مولدهای شامل مقدار اولیه، وجود دارند [۴۰-۳۶].

همان‌طور که شرح داده شد، در مولدهای دارای مقدار اولیه، مقدار اولیه استفاده‌شده باید دارای توزیع یکنواخت بوده و به نظر می‌رسد این یک تناقضی[۴۹] ایجاد می‌کند چون هدف ما از طراحی این مولدها تولید یک دنباله با توزیع نزدیک به توزیع یکنواخت است و بدون داشتن یک مولد اعداد تصادفی یک مقدار اولیه با توزیع یکنواخت را چگونه تولید کنیم. ولی این محدودیت ذکرشده برای مقدار اولیه استفاده‌شده در این مولدها زیاد جدی به نظر نمی‌رسد. در بسیاری از مولدهای صریح[۵۰]، طول مقدار اولیه استفاده‌شده به‌صورت تابع لگاریتمی از طول رشته بیت ورودی مولد است. برای مقادیر به‌اندازه کافی کوچک از d، نیاز به تصادفی بودن مقدار اولیه را می‌توان با شمارش کامل[۵۱] تمام 2^d دنباله ممکن برای مقدار اولیه جایگزین نمود. در الگوریتم‌های تصادفی شده، این شمارش با استفاده از روش رأی‌گیری تجمعی[۵۲] انجام می‌شود که این امکان را می‌دهد تا یک منبع یکنواخت را به‌خوبی شبیه‌سازی کنیم. اما برای کاربردهای رمزنگاری این روش معتبر نیست چون ویژگی غیرقابل‌پیش‌بینی بودن[۵۳] را ندارد [۴۱].

در مولدهای اعداد تصادفی کوانتومی، مولدهای دارای مقدار اولیه باعث می‌شوند مولد در برابر حملات بیرونی امن باشد. ساختارها و طرح‌هایی وجود دارند که اثبات‌هایی برای مقاوم بودن آن‌ها در برابر حملات کوانتومی[۵۴] با توان محاسباتی مختلف ارائه‌شده است [۴۲].

۲-۲-۱-معرفی مقدماتی مولد ترویسان

اولین مولد جالب‌توجه، مولد ترویسان[۵۵] بوده که در قسمت سوم در مورد آن مفصل بحث می‌کنیم. این مولد برخی ویژگی‌های جالب مانند مقاوم بودن در برابر حملات کوانتومی و روش جدید استفاده‌شده جهت حفظ نمودن تصادفی بودن مقدار اولیه را دارا است. مولد ترویسان بر اساس مولد اعداد شبه تصادفی نیسان-ویدگرسون[۵۶] طراحی‌شده است. مولد ترویسان را به‌صورت تابع تصادفی می‌توان در نظر گرفت که جدول صحت[۵۷] آن به‌وسیله بیت‌های حاصل از یک منبع ضعیف ارائه می‌شود. این تابع تصادفی، d بیت حاصل از یک مقدار اولیه با توزیع یکنواخت را در PRNG و حالت مولد بسط می‌دهد. مولدهای دیگری با استفاده از مولد ترویسان طراحی‌شده‌اند و برای استفاده با مولدهای اعداد تصادفی کوانتومی در توزیع کلید کوانتومی پیاده‌سازی شده‌اند. مزیت اصلی این روش‌ها این است که اندازه مقدار اولیه استفاده‌شده در آن‌ها به‌صورت تابع چند لگاریتمی از اندازه بلوک‌های ورودی است اما در پیاده‌سازی‌های عملی این روش‌ها، فرآیند تولید بیت به‌کندی صورت می‌گیرد چون به محاسبات تقریباً زیادی نیاز است تا خروجی تولید شود [۴۹-۴۳].

۲-۲-۲-معرفی مقدماتی روش درهم‌ساز دو-فراگیر

دومین مولد جالب‌توجه که یک روش جالب با ویژگی‌های منحصربه‌فرد است روش درهم‌ساز دو-فراگیر بوده که در قسمت سوم شرح داده می‌شود. لم درهم‌ساز باقی‌مانده[۵۸]،که در قسم سوم آورده شده، نشان می‌دهد که خروجی یک تابع درهم‌ساز دو-فراگیر[۵۹] حاصل از ورودی با آنتروپی به انداز کافی زیاد، تقریباً به‌صورت خروجی تصادفی با توزیع یکنواخت است. با استفاده از توابع درهم‌ساز دو-فراگیر، می‌توان اعداد تصادفی از منابع ضعیف تولید کرد به‌نحوی‌که در برابر حمله شنود، امن هستند. اگر یک تخمین خوب یا یک کران محافظه‌کارانه[۶۰] از همبستگی منبع تصادفی ضعیف با استراق سمع کننده داشته باشیم، در این صورت، با استفاده از آنتروپی شرطی، می‌توان از تعمیم لم درهم‌ساز باقی‌مانده با اطلاعات جانبی[۶۱] استفاده نمود. در اغلب موارد این اطلاعات جانبی می‌تواند کوانتوم نیز باشد. در مولد اعداد تصادفی کوانتومی با نویز فنی[۶۲]، می‌توانیم فرض کنیم که تمام میزان تصادفی بودنی که از غیر کامل بودن سیستم کوانتومی ناشی شده و یا از ناسازگار بودن با سیستم کوانتومی تولید بیت‌های تصادفی ناشی می‌شود، به علت وجود استراق سمع کننده است. در چنین شرایطی نیز می‌توان یک مولد دارای مقدار اولیه طراحی نمود که خروجی‌هایی با توزیع تقریباً یکنواخت تولید کند و خروجی آن مستقل از سامانه‌های تأثیرگذار بیرونی باشد. این روش‌ها در توزیع کلید کوانتومی برای تقویت امنیت و حریم خصوصی طرح‌ها استفاده‌شده‌اند [۶۱-۵۰].

تولید اعداد تصادفی با استفاده از توابع درهم‌ساز دو-فراگیر یا توابع درهم‌ساز l-فراگیر ما را مجبور می‌کند تا از مقدار اولیه‌ای استفاده کنیم که طول آن‌ها در مقایسه با اندازه بلوک، یعنی n، بزرگ هستند. نکته مثبتی که وجود دارد این است که این مقدار اولیه‌ها می‌توانند دوباره نیز مورداستفاده قرار گیرند. یک مقدار اولیه‌ای که به‌صورت تصادفی انتخاب‌شده و دارای توزیع یکنواخت عمومی بوده می‌تواند دوباره استفاده شود و با استفاده از آن می‌توان مولد دارای مقدار اولیه‌ای طراحی کرد که با منبع تصادفی غیر کامل و در حضور دشمن، امن است [۶۵-۶۲].

۲-۲-۳-مقایسه سرعت پیاده‌سازی مولدهای اعداد تصادفی

در مقایسه با مولد ترویسان، سرعت پیاده‌سازی توابع درهم‌ساز دو-فراگیر یا توابع درهم‌ساز l-فراگیر، بهتر بوده و از منابع محاسباتی کمتری نیز استفاده می‌کند ولی از مقدار اولیه‌هایی با طول بزرگ‌تر استفاده می‌کند. برخی پیاده‌سازی‌ها، مانند درهم‌سازی با ماتریس‌های دودویی تصادفی توپلیتز[۶۳]، دارای سرعت بهتری هستند که در بخش بعد مفصل در مورد نحوه ساخت آن بحث می‌شود. در چنین مولدهایی، مقدار اولیه استفاده‌شده به‌صورت یک ماتریس مستطیلی است. این ماتریس در یک بردار دودویی از اندازه n که توسط منبع تولیدشده، ضرب می‌شود تا خروجی مولد تولید شود. بیت‌های حاصل از خروجی این مولد تقریباً مستقل هستند. این روش در برخی دستگاه‌های تجاری که شامل توابع مولد هستند مورداستفاده قرار می‌گیرد. در این دستگاه‌ها یک ماتریس تصادفی وجود دارد که به‌عنوان مقدار اولیه عمل می‌کند و به‌صورت کدگذاری شده در داخل دستگاه توزیع‌شده است. در این دستگاه‌ها، متقاعد نمودن کاربر از تصادفی بودن و یکنواخت بودن مقدار اولیه استفاده‌شده، کار بسیار سختی بوده ولی لازم است که فقط یک‌بار این کار انجام شود. روش‌های پیچیده طولانی، مانند جمع دودویی مکرر خروجی‌های چندین مولد مستقل، روش‌های قابل قبولی برای تولید اعداد تصادفی هستند [۶۹-۶۶].

به‌تازگی برای پیاده‌سازی مولدهای اعداد تصادفی خانواده متنوعی از توابع درهم‌ساز رمزنگاری مانند الگوریتم درهم‌ساز امن (SHA[۶۴]) استفاده‌شده و همچنین درهم‌ساز مبتنی بر سیستم رمز AES[65] بکار برده شده است. این الگوریتم‌ها با دقت بالا طراحی‌شده‌اند و دارای میزان تصادفی بودن بالایی هستند. ولی اغلب توابع درهم‌سازی رمزنگاری توابع پیچیده‌ای بوده و نیاز به منابع محاسباتی بالایی دارند. این ویژگی برای مواردی که قصد طراحی مولد اعداد تصادفی با نرخ خروجی بالا داریم می‌تواند یک عامل محدودکننده باشد. اما باید توجه داشت همان‌طور که در بالا اشاره شد الگوریتم‌های جدید درهم‌ساز مانند SHA3، در تصادفی سازی داده خام، قدرت و عملکرد بسیار خوبی از خود نشان داده‌اند. درواقع یک بده و بستان بین میزان موفقیت در آزمون‌های آماری و پیچیدگی پیاده‌سازی طرح‌های پساپردازش وجود دارد [۷۱-۷۰].

۳-معرفی دو روش پساپردازش Toeplitz و Trevisan برای مولدهای اعداد تصادفی کوانتومی

مولدهای اعداد تصادفی کوانتومی (QRNG) اصولاً می‌توانند ابزاری برای تولید اعداد تصادفی قابل‌اثبات از نظر تئوری، ارائه دهند. در عمل، متأسفانه خروجی تصادفی کوانتومی به دلیل نویزهای کلاسیک به ناچار با تصادفی کلاسیک مخلوط می‌شود. برای تقطیر خروجی تصادفی کوانتومی، باید در ابتدا تصادفی بودن منبع را کمی نموده و همچنین یک استخراج‌کننده تصادفی اعمال گردد. در این قسمت، یک چارچوب عمومی برای ارزیابی خروجی QRNGهای واقعی با استفاده از اصل حداقل آنتروپی معرفی‌شده و در ادامه، این چارچوب برای دو سیستم اعداد تصادفی کوانتومی شناخته‌شده اعمال می‌گردد.

۳-۱-اهمیت استفاده از مولدهای اعداد تصادفی کوانتومی

اعداد تصادفی نقش مهمی در بسیاری از زمینه‌های علم، فناوری و صنعت مانند رمزنگاری، آمار، شبیه‌سازی‌های علمی و قرعه‌کشی، ایفا می‌کنند. مولدهای اعداد شبه تصادفی (PRNG[۶۶]) بر اساس پیچیدگی‌های محاسباتی[۶۷] به‌خوبی در چند دهه گذشته توسعه‌یافته‌اند و می‌توانند اعداد تصادفی با سرعت‌بالا را با هزینه کمی تولید کنند. بااین‌حال، اشکال اصلی PRNG ها این است که تصادفی بودن خروجی تولیدشده، از لحاظ نظری قابل‌اثبات نیست. درواقع، همه PRNG-های (مبتنی بر نرم‌افزار) را می‌توان با استفاده از یک الگوریتم قطعی[۶۸] و در اختیار داشتن توان محاسباتی کافی، تحلیل و بررسی نمود. این مولدهای شبه تصادفی در بسیاری از کاربردها مانند موارد رمزنگاری مشکل جدی در امنیت ایجاد می‌کند و مهاجم قادر است با استفاده از این ضعف، شنود و یا جعل و یا در بدترین حالت، کلید رمزنگاری

را کشف نماید. به‌تازگی، مایکروسافت تائید کرده است که برخی سیستم‌عامل‌های این شرکت حاوی اشکالات[۶۹] به علت استفاده از RNG است. همچنین، برخی نقص‌های امنیتی در روش‌های رمزگذاری آنلاین به دلیل نقص[۷۰] در تولید اعداد تصادفی به وجود آمده است.
برای حل مشکلات امنیتی به وجود آمده توسط PRNG–ها، RNG های فیزیکی[۷۱] توسعه‌یافته‌اند. به‌طور خاص، ماهیت احتمالی[۷۲] مکانیک کوانتومی یک راه طبیعی برای ساختن یک RNG را ارائه می‌دهد که از نظر تئوری نیز قابل‌اعتماد بوده و آن را مولدهای اعداد تصادفی کوانتومی (QRNG) می‌نامیم. توجه داشته باشید که برخی از RNGهای فیزیکی در ریزپردازنده‌ها[۷۳] گنجانده‌شده‌اند، اگرچه که خروجی‌های تصادفی تولیدشده، ماهیت مکانیکی کوانتومی ندارد. در تئوری، یک QRNG می‌تواند اعداد تصادفی با خروجی تصادفی قابل‌اثبات را تولید نماید. در عمل، سیگنال‌های کوانتومی[۷۴] (منبع خروجی‌های تصادفی واقعی[۷۵]) ناگزیر با نویزهای کلاسیک[۷۶] مخلوط می‌شوند. یک دشمن یا شخص ثالث[۷۷] (Eve) در اصل می‌تواند نویز کلاسیک را کنترل نموده و اطلاعات جزئی[۷۸] در مورد اعداد تصادفی خام به دست آورد. در این فاز، دستگاه قابل‌اعتماد[۷۹] فرض می‌شود، اما باید توجه داشت گه اگر سیستم با دقت بیشتری کالیبره[۸۰] شود آنگاه نویز کلاسیک ممکن است قطعی[۸۱] باشد. به‌عنوان‌مثال، فرض کنید که ورودی دستگاه یک منبع تغذیه خارجی است.آنگاه از طریق نظارت دقیق بر مقدار ورودی برق به دستگاه ، می‌توان مشخص نمود که برق ممکن است چند درصد در نوسان باشد. در اصل، منبع چنین نوسانات یک منبع تغذیه خارجی ممکن است اقدام یک Eve باشد کسی که اطلاعات کاملی در مورد ارزش واقعی منبع تغذیه در هر زمان دارد. بنابراین، لازم است از یک روش پس پردازش برای تقطیر خروجی تصادفی واقعی استفاده شود که Eve تقریباً هیچ اطلاعاتی در مورد آن ندارد. این روش تقطیر[۸۲] استخراج تصادفی[۸۳] نامیده می‌شود که با استفاده از استخراج‌کننده‌های[۸۴] تصادفی انجام می‌شود. به‌عبارت‌دیگر از استخراج‌کننده‌های تصادفی برای تقطیر خروجی تصادفی واقعی و از بین بردن اثر نویزهای کلاسیک استفاده می‌شود. هدف استخراج‌کننده‌های تصادفی، استخراج تصادفی (تقریباً) کامل از داده‌های خام[۸۵] تولیدشده از یک QRNG عملی با کمک یک مقدار اولیه تصادفی کوچک[۸۶] است که به منبع تصادفی اضافی نیاز دارد. پارامتر کلیدی ورودی یک استخراج‌کننده تصادفی مربوط به اصل حداقل آنتروپی[۸۷] داده‌های خام است. بااین‌وجود، یک روش کلی برای تعیین کمیت[۸۸] حداقل آنتروپی از داده‌های خام یک QRNG هنوز وجود ندارد.

برای استخراج مقادیر تصادفی، قبلاً از روش‌های ساده‌ای برای QRNG ها استفاده می‌گردید. برای نمونه یکی از روش‌های رایج استفاده از عملیات XOR بوده که در این روش در ابتدا داده‌های خام را به دو رشته بیت تقسیم نموده و در ادامه یک عملیات XOR بیتی[۸۹] بین آن‌ها انجام می‌شود. همچنین، روش‌هایی مانند انجام عملیات بر روی کم‌ارزش‌ترین بیت[۹۰] و یا استفاده از توابع درهم‌ساز[۹۱] پیشنهاد و پیاده‌سازی شده‌اند. این عملیات مطمئناً می‌تواند داده‌های خام را برای گذراندن برخی از آزمون‌های آماری برای رشته‌های تصادفی، اصلاح کند. بااین‌حال، نکته کلیدی این است که تصادفی بودن خروجی تولیدشده از نظر تئوری اطلاعات قابل‌اثبات نیست. به‌تازگی، یک روش استخراج تصادفی پیچیده‌تر[۹۲] پیشنهادشده که تصادفی بودن را با استفاده از آنتروپی شانون[۹۳] به‌جای اصل حداقل آنتروپی تعیین می‌کند و توابع درهم‌ساز را برای استخراج اعمال می‌نماید. متأسفانه، به دو دلیل تصادفی بودن خروجی استخراج‌شده در روش پیشنهادی هنوز از نظر تئوری اطلاعات قابل‌اثبات نیست. دلیل اول اینکه تصادفی بودن را نمی‌توان به‌خوبی با آنتروپی شانون تعیین نمود و دوم اینکه تصادفی نمودن با استفاده از توابع درهم‌ساز بر فرض‌های محاسباتی[۹۴] متکی است.

در مقابل، استخراج‌کننده تصادفی مهم و امیدوارکننده‌ای به نام استخراج‌کننده Trevisan، نه‌تنها دارای مزیت صرفه‌جویی در داده‌ها در مقایسه با سایر ساختارها بوده بلکه به دلیل ایمن بودن آن در برابر حملات کوانتومی، بسیار موردتوجه جدی پژوهشگران این حوزه قرارگرفته است. طول مقدار اولیه استخراج‌کننده Trevisan نسبت به طول ورودی چند لگاریتمی[۹۵] بوده و همچنین می‌توان ثابت نمود که استخراج‌کننده قوی‌ای نیز است. به‌عبارت‌دیگر می‌توان از مقدر اولیه تصادفی استخراج‌کننده Trevisan دوباره استفاده نمود. این موضوع بسیار مهم بوده زیرا برای یک تابع درهم‌ساز عمومی محبوب مانند Toeplitz، که به‌خوبی برای تقویت حریم خصوصی[۹۶] در توزیع کلید کوانتومی[۹۷] توسعه‌یافته، مقدار اولیه تصادفی مورداستفاده برای ساخت ماتریس در تابع درهم‌ساز Toeplitz طولانی‌تر از رشته خروجی است. این به این معنی است که اگر تابع درهم‌ساز مستقیماً برای استخراج خروجی تصادفی استفاده شود، هیچ خروجی تصادفی خالصی[۹۸] قابل‌استخراج نیست. لازم به ذکر است که استخراج‌کننده‌های تصادفی نیز می‌توانند برای تقویت حریم خصوصی در QKD استفاده شوند. توجه داشته باشید که تقویت حریم خصوصی یک گام مهم در پس پردازش QKD است. ثابت‌شده است که چند استخراج‌کننده تصادفی در برابر حمله کانال جانبی کوانتومی ایمن هستند. مزیت اصلی این است که هیچ ارتباط کلاسیکی برای تقویت حریم خصوصی موردنیاز نیست. همچنین، استفاده از تکنیک‌های توسعه‌یافته در استخراج تصادفی برای تقویت حریم خصوصی یک

موضوع تحقیقاتی جالب است. علیرغم ویژگی‌های تئوری قابل‌توجه استخراج‌کننده Trevisan، اجرای عملی آن هنوز وجود ندارد. این احتمالاً به این دلیل است که استخراج‌کننده Trevisan ساختار نسبتاً پیچیده‌ای دارد که ممکن است جامعه اطلاعات کوانتومی با آن آشنایی نداشته باشد و اجرای آن شامل بده بستان بین سرعت و مقادیر پارامترهای امنیتی است.

در این قسمت یک روش کلی برای کمی سازی تصادفی[۹۹] و اجرای عملی استخراج‌کننده Trevisan ارائه می‌شود که می‌تواند مشکلات[۱۰۰] بده بستان در این استخراج‌کننده را رفع نماید. درواقع، یک طرح کلی ارائه می‌شود که می‌تواند داده‌های خام را از یک QRNG به اعداد تصادفی که تقریباً از یک توزیع یکنواخت[۱۰۱] پیروی می‌کنند، پردازش نماید. دو قسمت مهم ارائه‌شده در این بخش عبارت‌اند از:

۱. در ابتدا یک چارچوبی برای کمی نمودن خروجی تصادفی کوانتومی از یک QRNG با استفاده از اصل حداقل آنتروپی ارائه می‌شود و سپس در مورد اینکه چگونه می‌توان این حداقل آنتروپی را در یک دستگاه فیزیکی ارزیابی نمود، بحث می‌شود. در گام بعدی، چارچوب پیشنهادشده برای دو QRNG مطرح در زمینه تولید اعداد تصادفی، اعمال می‌گردد.

۲. پیاده‌سازی اولیه[۱۰۲] از استخراج‌کننده Trevisan ارائه می‌گردد و نشان داده می‌شود که چگونه می‌توان از چنین پیاده‌سازی در QRNG های واقعی[۱۰۳] استفاده نمود. با استفاده از این پیاده‌سازی، مرحله محاسباتی اصلی[۱۰۴] آشکار می‌گردد که باعث محدودیت در سرعت استخراج‌کننده Trevisan می‌گردد.

بر اساس چند فرض قابل‌قبول و مناسب[۱۰۵] در مدل فیزیکی QRNG، تصادفی بودن خروجی استخراج‌شده از پس پردازش پیشنهادی، از نظر تئوری اطلاعات قابل‌اثبات است. طرح عمومی پس پردازش پیشنهادی شامل سه مرحله زیر است:

- مدل‌سازی و مشخص نمودن[۱۰۶] نحوه راه‌اندازی[۱۰۷] QRNG از طریق اندازه‌گیری[۱۰۸]
- کمی نمودن خروجی تصادفی کوانتومی از داده‌های خام با استفاده از اصل حداقل آنتروپی
- اعمال نمودن یک استخراج‌کننده تصادفی

همان‌طور که در قبل نیز اشاره شد برای نشان دادن کلیت روش معرفی‌شده در این قسمت، طرح معرفی‌شده بر روی دو نوع مختلف از QRNG های واقعی اعمال‌شده است.

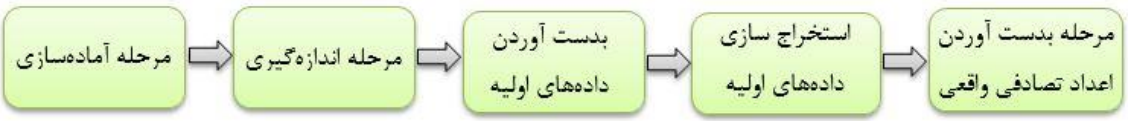
۳-۲-ارزیابی خروجی‌های تصادفی به‌دست‌آمده از روش‌های کوانتومی

در این بخش، یک چارچوب کلی برای ارزیابی خروجی‌های تصادفی (واقعی) به‌دست‌آمده از یک QRNG عملی ارائه می‌شود. مولدهای اعداد تصادفی کوانتومی که به‌تازگی توسعه داده‌شده‌اند به‌عنوان نمونه‌هایی از فرآیند ارزیابی موردبحث قرار می‌گیرند. لازم به ذکر است که روش ارزیابی پیشنهادشده در این قسمت، یک روش عمومی بوده که می‌تواند برای سایر QRNG ها با تغییرات خاصی اعمال گردد.

۳-۲-۱-مدل فیزیکی

به‌طورکلی، اعداد تصادفی تولیدشده توسط یک QRNG از یک اندازه‌گیری خاص[۱۰۹]، به دست می‌آیند. نتیجه اندازه‌گیری شده به‌عنوان سیگنال کوانتومی[۱۱۰] در نظر گرفته می‌شود. این سیگنال کوانتومی ناگزیر[۱۱۱] با نویزهای کلاسیک مانند تشخیص پس‌زمینه[۱۱۲] و نویزهای الکترونیکی[۱۱۳] مخلوط می‌شود. از دیدگاه یک تحلیل‌گر رمزنگاری، این نویزهای کلاسیک ممکن است در بدترین حالت برای دشمن شناخته‌شده باشند و یا حتی دست‌کاری[۱۱۴] شوند. بنابراین، هدف اصلی از انجام پس پردازش برای یک QRNG این است که در مرحله اول خروجی‌های (واقعی) تصادفی به‌دست‌آمده از روش‌های کوانتومی را استخراج نموده و در گام بعدی مشارکت نویزهای کلاسیک را حذف نماید.

فرایند اجرای یک مولد اعداد تصادفی کوانتومی در شکل (۱) نشان داده‌شده است.



شکل (۱): نمودار کلی از راه‌اندازی مولد اعداد تصادفی

در شکل (۱)، ابتدا یک حالت کوانتومی[۱۱۵] تهیه می‌شود که منبع خروجی‌های تصادفی واقعی است. سپس، یک اندازه‌گیری بر روی حالت کوانتومی انجام می‌شود. درنهایت، داده‌های خام یا اولیه توسط یک استخراج‌کننده تصادفی پس پردازش می‌شوند تا اعداد واقعاً تصادفی[۱۱۶] تولید شوند. برای مثال حالت کوانتومی را در نظر بگیرید که در آن فازهای تصادفی فوتون‌ها را از گسیل‌های[۱۱۷] خود به خودی[۱۱۸] مشخص می‌کند که با استفاده از یک لیزر در نزدیک سطح آستانه[۱۱۹] آن تهیه می‌گردد. اندازه‌گیری توسط یک سیستم خود-هترودایین[۱۲۰] تأخیری[۱۲۱] انجام می‌شود و داده‌های خام بر اساس یک مدل فیزیکی ارزیابی‌شده و توسط استخراج‌کننده‌های تصادفی پردازش می‌شوند. برای مثال در برخی از QRNG-ها حالت کوانتومی توسط دامنه مربعات[۱۲۲] حالت خلاء[۱۲۳] ایجادشده و اندازه‌گیری توسط یک آشکارساز هم‌آمیز[۱۲۴] انجام می‌گردد. در ادامه داده‌های خام توسط یک استخراج‌کننده از نوع توابع درهم‌ساز[۱۲۵]، پردازش می‌شوند.

۳-۲-۲-روش‌های ارزیابی خروجی تصادفی کوانتومی

پارامتر کلیدی که در این قسمت باید ارزیابی کنیم، اصل حداقل آنتروپی بوده و مربوط به سیگنال کوانتومی موجود در داده‌های خام یا اولیه است. در ادامه روشی برای ارزیابی حداقل آنتروپی با استخراج نمودن توزیع احتمال سیگنال کوانتومی ارائه می‌شود. به‌عنوان اولین مثال برای نشان دادن رویه‌های دقیق[۱۲۶] از فرایند ارزیابی، تنظیمات QRNG را انجام می‌دهیم. در این حالت، سیگنال کوانتومی از نوسانات خلاء[۱۲۷] می‌آید و سایر منابع نوسانات فاز[۱۲۸] به‌عنوان نویز کلاسیک تعریف می‌گردد.

فرض‌های در نظر گرفته‌شده برای مدل فیزیکی موردنیاز برای استخراج توزیع احتمال سیگنال کوانتومی عبارت است از:

۱. سیگنال کل[۱۲۹] که ترکیبی از سیگنال کوانتومی و نویز کلاسیک است. همچنین، سیگنال کوانتومی مستقل از نویز کلاسیک است.
۲. سیگنال کوانتومی بر اساس توزیع گاوسی[۱۳۰] است. کل سیگنال آنالوگ توسط یک تبدیل آنالوگ به دیجیتال[۱۳۱] (ADC) دیجیتالی می‌شود.
۳. نسبت بین واریانس سیگنال کوانتومی و نویز کلاسیک را می‌توان تعیین نمود که با γ نشان داده می‌شود.
٤. واریانس کل سیگنال را می‌توان با نمونه‌برداری مشخص نمود که با σ^2_{total} نشان داده می‌شود. توجه داشته باشید که آخرین فرض، زمانی می‌تواند برآورده شود که دنباله داده‌های خام، مستقل و به‌طور یکسان توزیع‌شده باشد. لازم به تذکر است نیازی به این فرض نیست که نویز کلاسیک از توزیع گاوسی پیروی نماید.

به‌طور خلاصه، برای استخراج توزیع احتمال سیگنال کوانتومی، نکته کلیدی یافتن واریانس آن است. این کار با اندازه‌گیری واریانس کل داده‌های خام و نسبت کوانتومی به کلاسیک[۱۳۲] انجام می‌شود. به‌عبارت‌دیگر با فرض‌های ۱، ۳ و ۴ می‌توان به‌راحتی واریانس کوانتومی را به‌صورت زیر استخراج نمود:

$$\sigma^2_{quantum} = \frac{\gamma \sigma^2_{total}}{1+\gamma} \qquad (۹)$$

از مقدار واریانس به همراه فرض در نظر گرفته‌شده در توزیع گاوسی، می‌توان کل توزیع احتمال سیگنال کوانتومی را به دست آورد. ازآنجایی‌که سیگنال آنالوگ توسط یک ADC هشت بیتی[۱۳۳] برای تولید بیت‌های دیجیتال، نمونه‌برداری می‌شود و با توجه به توزیع گاوسی سیگنال کوانتومی، می‌توان توزیع احتمال خروجی دیجیتالی شده[۱۳۴] را روی $\{0,1\}^8$ ارزیابی نمود. در ادامه، حداقل آنتروپی سیگنال کوانتومی را می‌توان با استفاده از تعریف ۱ به دست آورد. با به‌کارگیری روش‌های محاسبه دقیق، حداقل آنتروپی ۶.۷ بیت در هر نمونه اولیه هشت بیتی (از یک ADC هشت بیتی) به دست می‌آید. به‌تازگی از ADC برای قطع کردن یکنواخت[۱۳۵] bins در امتداد محور ولتاژ[۱۳۶] استفاده می‌شود. با طراحی دقیق اندازه bin، می‌توان حداقل آنتروپی را افزایش داد.

نوسان حالت خلاء مورداستفاده، از توزیع گاوسی تبعیت می‌کند بنابراین فرض‌های در نظر گرفته‌شده برای یک مدل فیزیکی، مشابه مواردی است که در بالا توضیح داده شد. برای تعیین دقیق کمیت میزان تصادفی بودن با استفاده از حداقل آنتروپی، از داده‌های مربوط به واریانس سیگنال کل و نویز کلاسیک استفاده می‌شود و سپس واریانس سیگنال کوانتومی استخراج می‌گردد. در ادامه به‌جای محاسبه

آنتروپی شانون، حداقل آنتروپی سیگنال کوانتومی را می‌توان با استفاده تعریف ۱ به دست آورد که این محاسبه با توجه به توزیع گوسی از سیگنال کوانتومی بر روی توزیع احتمال خروجی دیجیتال شده در $\{۰,۱\}$ انجام می‌شود.
قابل‌ذکر است که روش پیشنهادشده در این قسمت می‌تواند برای انواع دیگر QRNGها نیز اعمال شود. برای مثال می‌توان این روش را برای مولد اعداد تصادفی که دارای متغیرهای گسسته هستند و نه متغیرهای پیوسته، اعمال نمود. نکته کلیدی در این قسمت، ارزیابی حداقل آنتروپی سیگنال‌های کوانتومی از طریق جداسازی کمی[۱۳۷] مشارکت[۱۳۸] آن از سیگنال‌های کوانتومی و نویزهای کلاسیک است که توسعه این مسئله می‌تواند یک موضوع تحقیقاتی جالب‌توجه در آینده باشد.

۳-۲-۳-ارائه یک کران بالا برای خروجی‌های تصادفی

تصادفی بودن خروجی یک طرح QRNG یک مسئله با محدودیت‌های خاصی است. این را می‌توان با ارائه کران بالای تصادفی، برای مثال از طریق آنتروپی شانون نشان داد که این پارامتر را می‌توان با استفاده از نتیجه اندازه‌گیری، استخراج نمود. همچنین کران بالا نشان می‌دهد که چقدر حاشیه[۱۳۹] برای بهبود بیشتر در پس پردازش باقی‌مانده است.
با یک مثال نشان می‌دهیم که چگونه می‌توان حد بالای آنتروپی[۱۴۰] را برای یک راه‌اندازی عملی یک QRNG، ارزیابی نمود. سیگنال کوانتومی توسط یک آشکارساز نوری[۱۴۱] (PD) اندازه‌گیری می‌شود. با توجه به یک آشکارساز کامل تفکیک عدد فوتون[۱۴۲]، حد بالای حداقل آنتروپی توسط عدد فوتون و در پنجره زمانی تشخیص[۱۴۳]، تعیین می‌گردد. توان لیزری که برای راه‌اندازی این مثال فرض می‌شود برابر با ۰.۹۵ میلی‌وات بوده که مربوط به فوتون‌های ۱.۵×۱۰^{۶} در ۱۵۵۰ نانومتر در یک پنجره زمانی تشخیص ۲۰۰ پیکو ثانیه است. بنابراین، حداکثر آنتروپی یک نمونه از PD را می‌توان با ۲۰.۵ بیت تخمین زد که حد بالایی از حداقل آنتروپی منبع QRNG است ($\log_{۲}(۱.۵\times۱۰^{۶})=۲۰.۵$).

۳-۳-استخراج نمودن مقادیر تصادفی به‌دست‌آمده

در این بخش، پیاده‌سازی‌های اولیه از استخراج‌کننده Trevisan و همچنین استخراج‌کننده تابع درهم‌ساز Toeplitz را شرح می‌دهیم که دو روش متداول و مهم پساپردازش هستند. این پیاده‌سازی قابل‌استفاده برای استخراج در QRNG های موجود است.

۳-۳-۱-استخراج‌کننده Trevisan

این استخراج‌کننده رویکردی را برای ساخت استخراج‌کننده‌های تصادفی بر اساس مولدهای اعداد شبه تصادفی پیشنهاد می‌کند. استخراج‌کننده Trevisan چندین مزیت نظری مهم دارد که به شرح زیر است. مزیت اول این است که در برابر یک دشمن با توان محاسباتی کوانتومی، ایمن است. مزیت دوم طول ورودی بوده که طول مقدار اولیه چند لگاریتمی است. مزیت سوم این است که می‌توان ثابت نمود با تغییرات خاصی در پارامترهای امنیتی، این استخراج‌کننده یک استخراج‌کننده قوی است. این در حالی است که پیاده‌سازی واقعی از این استخراج‌کننده مهم هرگز در تحقیقات مرتبط با این موضوع، گزارش نشده است.
دو مرحله اصلی برای ساخت استخراج‌کننده Trevisan شامل مراحل استخراج‌کننده یک بیتی[۱۴۴] و طراحی ترکیبی[۱۴۵] است. ساختار مرحله یک بیتی بر اساس یک کد تصحیح خطا[۱۴۶] بوده که این کد از الحاق[۱۴۷] یک کد رید-سولومون[۱۴۸] با یک کد هادامارد[۱۴۹]، ساخته‌شده است. برای پیاده‌سازی مرحله طراحی ترکیبی، از یک نسخه تصحیح‌شده[۱۵۰] از طراحی نیسان-ویگدرسون[۱۵۱]، استفاده‌شده است.
پیاده‌سازی این نسخه پیشنهادی از استخراج‌کننده Trevisan، دارای سرعت ۰.۷ کیلوبایت بر ثانیه است. تمرکز اصلی بر روی مرحله محاسباتی اصلی[۱۵۲] است که سرعت استخراج Trevisan را محدود می‌کند به‌طوری‌که این مرحله بر روی استخراج‌کننده یک بیتی[۱۵۳] مبتنی بر تصحیح خطا[۱۵۴] قرار دارد. یکی از مهم‌ترین بهبودهای این روش این است که به‌جای فهرست کردن کدهای تصحیح خطا از استخراج‌کننده‌های یک بیتی جایگزین استفاده نماییم که درنتیجه این تغییر، سرعت استخراج در پیاده‌سازی به‌طور قابل‌توجهی بهبود می‌یابد به‌طوری‌که می‌تواند تا ۱۵۰ کیلوبایت بر ثانیه برسد. در مقایسه با استخراج‌کننده چکیده ساز Toeplitz، استخراج‌کننده Trevisan به مقدار اولیه‌ای با طول کوتاه‌تر نیاز دارد. بنابراین، استخراج‌کننده Trevisan در مواردی که بیت‌های مقدار اولیه تصادفی محدود است، دارای مزیت خاصی است. اگرچه که نتایج پیاده‌سازی نشان می‌دهد سرعت استخراج‌کننده درهم‌ساز Toeplitz برابر با ۴۴۱ کیلوبایت

بر ثانیه بوده و بسیار سریع‌تر از استخراج‌کننده Trevisan است که سرعت پیاده‌سازی آن ۰.۷ کیلوبایت بر ثانیه هست اما هنوز به مسئله مقایسه دقیق بین این دو استخراج‌کننده، به‌طور دقیق پرداخته نشده است.

۳-۳-۱-۱-روش پیاده‌سازی

در این قسمت یک نمای کلی از روش پیاده‌سازی ارائه می‌دهیم. در ابتدا داده خام به‌دست‌آمده از یک QRNG که یک رشته بیت از طول n_i و دارای حداقل آنتروپی k بوده را به همراه یک مقدار اولیه تصادفی از طول d بیت (y) را به‌عنوان ورودی به یک استخراج‌کننده $(k,\varepsilon,n_i,d,n_f)$ می‌دهیم. این استخراج‌کننده ترکیبی از یک کد تصحیح‌خطا $(n_i,1/2-\varepsilon/4n_f)$ و یک طرح (m_e,ρ) بوده که خروجی آن یک رشته بیت با طول n_f هست که توزیع آن به توزیع یکنواخت ε-نزدیک است. پارامترهای k، ε و n_i با استفاده از منابع مرتبط با اعداد تصادفی به‌دست‌آمده است. لازم به ذکر است که پارامتر d را تا حد امکان کوچک و n_f تا حد امکان بزرگ باید تعیین نمود.

در مرحله اول رشته n_i بیتی را توسط کد تصحیح خطای $(n_i,1/2-\varepsilon/4n_f)$ به یک رشته $\bar{n}$ بیتی نگاشت می‌کنیم. در این قسمت $\bar{n}$ توانی از عدد ۲ است. در عمل، می‌توان کد رید-سولومون و کد هادامارد را باهم الحاق نمود به‌طوری‌که طول کدکلمه به‌صورت زیر شود:

$$\bar{n}=2^{2m_e},\quad m_e=\left\lceil \log_2 n_i+2\log_2 n_f-2\log_2\varepsilon+4\right\rceil \qquad (10)$$

همچنین n_f را می‌توان برای ساخت کد تصحیح خطا توسط k محدود نمود. حال نوبت به ساخت یک طرح (m_e,ρ) می‌رسد که پارامترهای آن به شرح زیر است:

$$m_e=\frac{1}{2}\log_2\bar{n}=O(\log_2(n_i/\varepsilon)),\quad \rho=[k-3\log_2(n_f/\varepsilon)-d-3]/n_f, \qquad (11)$$

پارامتر ρ در طرح را می‌توان به‌عنوان نسبت حداقل آنتروپی[۱۵۵] قابل‌استخراج در نظر گرفت. برای بهینه نمودن طول خروجی می‌توان به‌سادگی $\rho=1$ فرض نمود. مقدار اولیه استخراج‌کننده که طول آن d فرض می‌شود، از بلوک‌های مقادیر اولیه با طول توان دوم کوچک‌ترین توان[۱۵۶] دو که بزرگ‌تر از m_e بوده تشکیل‌شده است. در این قسمت علاقه‌مند هستیم که یک طرح با $\rho=1$ داشته باشیم که بتوانیم خروجی تصادفی بیشتری استخراج نماییم. تعداد این‌چنین بلوک‌هایی و به طبع آن طول مقدار اولیه عبارت است از:

$$\begin{aligned} b&=\left\lceil \log_2 n_f\right\rceil-m_d+1,\\ d&=2^{2m_d}b=O(\log_2^2(n_i/\varepsilon)\log_2 n_i),\\ m_d&\equiv\left\lceil \log_2 2m_e\right\rceil \end{aligned} \qquad (12)$$

قابل‌ذکر است که در این قسمت می‌توان هر طرحی که شرایط $d\geq\left\lceil \log_2\bar{n}\right\rceil^2 b$ و $\rho=1$ را داشته باشد بکار برد. همچنین، i-امین بیت از n_f بیت خروجی توسط y_{S_i} امین بین از رشته $\bar{n}$ بیتی کدگذاری شده، به دست می‌آید به‌طوری‌که y_{S_i} یک زیررشته[۱۵۷] از y بوده که توسط بیت‌هایی از y که موقعیت آن‌ها توسط عناصر S_i تعیین‌شده، تشکیل می‌شوند.

۳-۳-۲-درهم‌سازی فراگیر

به دلیل شباهت بین تعاریف استخراج‌کننده‌ها و تقویت حریم خصوصی، هر طرح تقویت حریم خصوصی[۱۵۸] را می‌توان به‌عنوان استخراج‌کننده استفاده نمود. بااین‌حال، یک تفاوت ظریف[۱۵۹] وجود دارد. در تقویت حریم خصوصی، مقدار اولیه تصادفی که به آن مقدار تصادفی عمومی گفته می‌شود، آزاد فرض می‌شود و این در حالی است که در یک استخراج‌کننده باید مقدار اولیه را در نظر گرفت زیرا در تشکیل بیت‌های تصادفی مشارکت دارد. بنابراین، استفاده مستقیم از طرح‌های تقویت حریم خصوصی، ممکن است برای استخراج مقادیر تصادفی کار نکند. درواقع، در تابع درهم‌ساز فراگیر معروفی مانند تابع درهم‌ساز Toeplitz، مقدار اولیه تصادفی مورداستفاده برای ساخت ماتریس Toeplitz طولانی‌تر از رشته خروجی است. این بدان معنی است که اگر تابع درهم‌ساز فراگیر مستقیماً برای استخراج

مقادیر تصادفی استفاده شود، هیچ مقدار تصادفی خالصی[۱۶۰] قابل‌استخراج نیست. برای غلبه بر این مشکل، باید ثابت نمود که طرح تقویت حریم خصوصی یک استخراج‌کننده قوی ایجاد می‌کند، بنابراین به شخص اجازه می‌دهد تا تصادفی بودن مقدار اولیه را برای کاربردهای بعدی[۱۶۱] حفظ نماید. نکته قابل‌توجه این است که می‌توان به‌راحتی توسط لم زیر ثابت نمود استخراج‌کننده‌های ساخته‌شده توسط توابع درهم‌ساز فراگیر، استخراج‌کننده‌های قوی هستند.

لم ۱: فرض کنید $H=\{h_1,h_2,\cdots,h_{2^d}\}$ یک خانواده از توابع درهم‌ساز فراگیر بوده به‌طوری‌که رشته بیت دودویی n بیتی را به رشته بیت دودویی m بیتی نگاشت می‌نماید. فرض کنید که X یک توزیع احتمال بر روی $\{0,1\}^n$ بوده که در شرط $H_\infty(X)\geq k$ صدق می‌نماید. آنگاه برای هر $x\in X$ و $h_y\in H$ که $y\in U_d$، توزیع احتمال تشکیل‌شده توسط $h_y(x)\circ y$ یک توزیع احتمال $\varepsilon=2^{(m-k)/2}$ نزدیک به توزیع یکنواخت U_{m+d} بوده که با استفاده از آن یک استخراج‌کننده قوی $(k,2^{(m-k)/2},n,d,m)$ تشکیل می‌شود.

یکی از نکاتی که از لم ۱ نتیجه می‌شود این است که ماتریس Toeplitz ممکن است در تقویت حریم خصوصی QKD مجدداً استفاده شود. سپس، می‌توان از یک کلید خصوصی (به‌عنوان مقدار اولیه) برای ساخت ماتریس Toeplitz برای تقویت حریم خصوصی بدون به خطر انداختن[۱۶۲] حریم خصوصی مقدار اولیه[۱۶۳] استفاده نمود. ازاین‌رو، استفاده مجدد از مقدار اولیه می‌تواند ارتباط کلاسیک را برای تقویت حریم خصوصی، که معمولاً در پساپردازش استاندارد QKD[164] موردنیاز است ذخیره نماید. همچنین برای تقویت حریم خصوصی، تقسیم نمودن داده کلید خام[۱۶۵] به بلوک‌های کوچک و اعمال یک ماتریس Toeplitz کوچک به‌صورت جداگانه، در عمل مفید است. بااین‌حال، اثر اندازه-محدود[۱۶۶] یک بلوک کوچک می‌تواند به‌طور قابل‌توجهی کارایی[۱۶۷] تقویت حریم خصوصی را کاهش دهد که این موضوع یک موضوع تحقیقاتی جالب است.

۳-۳-۳-معرفی روش Toeplitz

در این قسمت، از ماتریس‌های Toeplitz برای ساخت تابع درهم‌ساز فراگیر استفاده‌شده و استخراج‌کننده Toeplitz پیاده‌سازی می‌شوند. برای ساخت یک ماتریس $n\times m$ به شکل Toeplitz فقط نیاز است که مقادیر درایه‌های سطر اول و ستون اول ماتریس را داشته باشیم و سایر عناصر از طریق عناصر سطر اول و ستوان اول به دست می‌آید. بنابراین، تعداد کل بیت‌های تصادفی موردنیاز برای ساخت (یا انتخاب) یک ماتریس Toeplitz برابر $n+m-1$ است. روش استخراج تابع درهم‌ساز Toeplitz به شرح زیر است:

۱. با توجه به دو ویژگی داده خام از طول n که شامل داشتن حداقل آنتروپی k و پارامتر امنیتی ε است طول خروجی به‌صورت زیر تعیین می‌شود:

$$m=k-2\log_2\varepsilon. \qquad (۱۳)$$

۲. یک ماتریس Toeplitz با استفاده از $n+m-1$ مقدار اولیه تصادفی می‌سازیم.

۳. رشته بیت تصادفی استخراج‌شده[۱۶۸] با استفاده از ضرب داده‌های خام در ماتریس Toeplitz، به دست می‌آید.

لازم به ذکر است که حداقل آنتروپی داده‌های خام با نرخ ۶.۷ بیت در هر نمونه[۱۶۹] محدود می‌شود. حال اگر طول رشته بیت ورودی $2^{12}=4096$ شود آنگاه طول رشته بیت خروجی $4069\times 6.7/8\geq 3430$ می‌شود. بنابراین، ما از یک ماتریس 4096×3230 به فرم Toeplitz برای استخراج مقادیر تصادفی استفاده می‌کنیم که با استفاده نمودن از این ماتریس و بکار بردن معادله (۱۳) نتیجه $\varepsilon<2^{-100}$ به دست می‌آید.

لازم به ذکر است که به‌تازگی ماتریس تابع درهم‌ساز Toeplitz برای تقویت حریم خصوصی QKD با اندازه بلوک بیش از ۱۰۶ پیاده‌سازی شده است. همان‌طور که در بالا بحث شد، تقویت حریم خصوصی به دلیل تأثیر اندازه محدود کلید، به بلوکی با اندازه بزرگ نیاز دارد و این در حالی است که در کاربردهای استخراج مقادیر تصادفی، اندازه بلوک کوچک، فقط کارایی را کاهش می‌دهد.

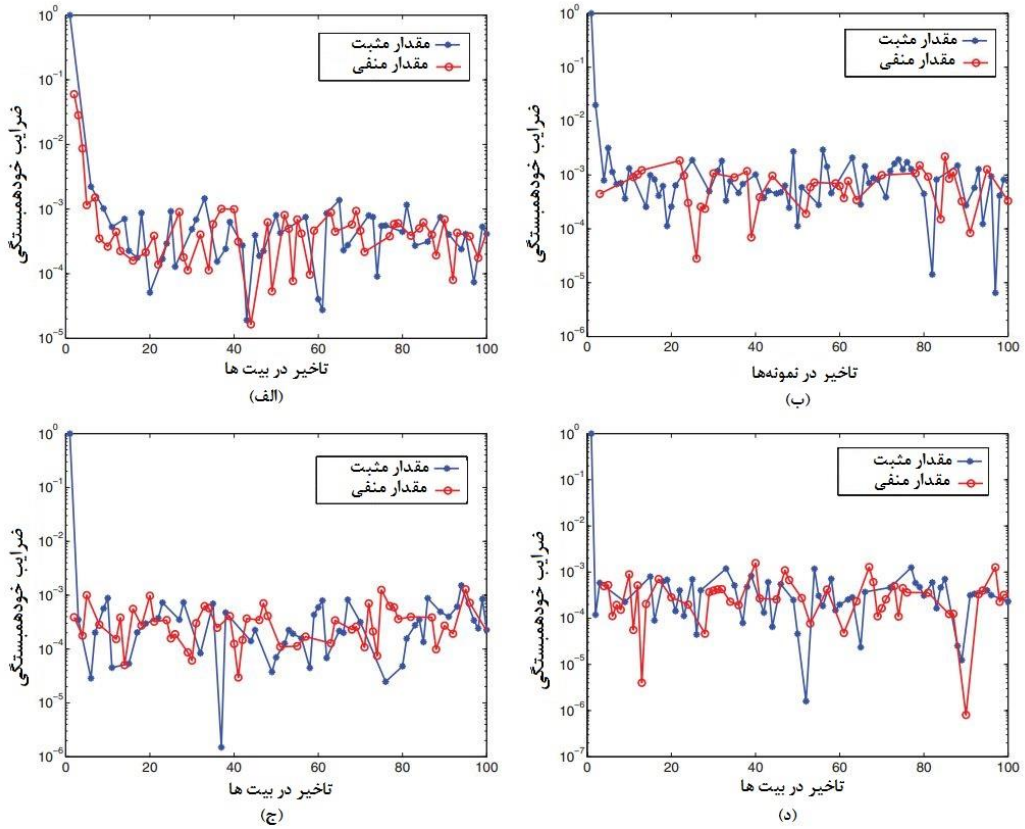
۳-۳-۴-انجام آزمون‌های ارزیابی مقادیر تصادفی

۳-۳-۴-۱-آزمون‌های آماری

در این قسمت، از آزمون‌های آماری استاندارد NIST برای ارزیابی نتایج استفاده‌شده است. ازآنجایی‌که، فقط داده‌های مربوط به یک QRNG متداول را داریم، از آن به‌عنوان یک آزمایش برای پیاده‌سازی استخراج‌کننده‌های تصادفی معرفی‌شده در این قسمت، استفاده می‌گردد. در ابتدا مشاهده شد که داده‌های خام QRNG به دلیل نویزهای کلاسیک مخلوط شده در داده‌های خام و همچنین این واقعیت که سیگنال‌های کوانتومی به‌دست‌آمده به‌جای توزیع یکنواخت، از یک توزیع گاوسی پیروی می‌کنند، آزمون‌های آماری را نتوانستند پاس نمایند. به همین دلیل، چندین مرتبه تابع درهم‌ساز Toeplitz و همچنین استخراج‌کننده Trevisan را بر روی داده‌های خام بکار بردیم. خروجی‌های هر دو استخراج‌کننده با موفقیت تمام آزمون‌های آماری استاندارد را پشت سر گذاشتند که این موضوع نشان می‌دهد پس پردازش پیشنهادشده در این قسمت، در استخراج مقادیر تصادفی با توزیع یکنواخت از یک منبع تصادفی ضعیف، مؤثر است.

۳-۳-۴-۲-آزمون خودهمبستگی

یک رویکرد جایگزین برای تائید تصادفی بودن مقادیر به‌دست‌آمده، ارزیابی خودهمبستگی است. آزمون خودهمبستگی داده‌های خام بین بیت‌ها در شکل (۲-الف) و بین نمونه‌ها در شکل (۲-ب) آمده است.



شکل (۲): نتایج ارزیابی خودهمبستگی

در شکل (۲) همبستگی نرمال شده[۱۷۰] با استفاده از یک رکورد ۱۰ مگابایتی از داده‌های خام ارزیابی‌شده است. شکل (۲-الف) مربوط به خودهمبستگی بین بیت‌ها در داده‌های خام است. مقدار ضریب خودهمبستگی برابر با ۹.۵×۱۰^{-۴} است. مهم‌ترین دلیل خودهمبستگی بیت بیت‌ها به علت استفاده از ADC هشت بیتی است.

شکل (۲-ب) مربوط به ضرایب خودهمبستگی در داده‌های خام در بین نمونه‌ها است. مقدار میانگین برابر با ۴.۹×۱۰^{-۴} است. شکل (۲-ج) مربوط به ضرایب خودهمبستگی حاصل از استخراج‌کننده تابع درهم‌ساز Toeplitz است. مقدار میانگین برابر با -۱.۰×۱۰^{-۵} است. شکل (۲-د) مربوط به ضرایب خودهمبستگی حاصل از نتایج استخراج‌کننده Trevisan است. مقدار میانگین برابر با ۱.۶×۱۰^{-۵} است. از لحاظ تئوری، برای یک رشته واقعاً تصادفی ۱۰×۱۰^{۶} بیتی، متوسط ضریب همبستگی نرمال شده با انحراف معیار استاندارد ۴×۱۰^{-۴} برابر با صفر است.

از شکل (۲-الف) می‌توان مشاهده نمود که خودهمبستگی فقط در یک نمونه هشت بیتی معنادار است اما در مجاورت[۱۷۱] و زیر آن برابر با ۱×۱۰^{-۳} است. همچنین مقادیر کم خودهمبستگی بین نمونه‌ها در شکل (۲-ب) این فرض را تائید می‌کند که دنباله داده‌های خام، دنباله‌ای از متغیرهای تصادفی، مستقل با توزیع یکسان[۱۷۲] است. لازم به ذکر است که به دلیل پهنای باند محدود یک آشکارساز عملی[۱۷۳] و همچنین نوسانات آماری[۱۷۴]، خودهمبستگی در حدود ۱×۱۰^{-۳} است اما هرگز به صفر نمی‌رسد.

بعد از انجام مرحله پساپردازش توسط یکی از دو استخراج‌کننده Travisan و یا تابع درهم‌ساز Toeplitz، نه‌تنها همبستگی در هشت بیت (از نمونه دیجیتالی شده توسط یک ADC هشت بیتی) حذف می‌شود، بلکه خودهمبستگی بیشتر از هشت بیت نیز به ۱×۱۰^{-۵} کاهش می‌یابد. خودهمبستگی خروجی‌های بعد از مرحله پساپردازش در شکل (۲-ج) و (۲-د) نشان داده‌شده است. با استفاده از مقادیر کم باقیمانده[۱۷۵] در این دو شکل می‌توان نتیجه گرفت که تصادفی بودن نتایج بعد از مرحله استخراج سازی، به‌خوبی انجام‌شده است.

۴-روش پساپردازش فون‌نیومان برای تولید اعداد تصادفی

در این قسمت یک بهبود و پیاده‌سازی از روش پساپردازشی فون‌نیومان n-بیتی ($\mathrm{VN_n}$) ارائه می‌گردد که روشی برای تولید دنباله‌ای از بیت‌های تصادفی غیر اریب (فاقد بایاس) از بیت‌های دارای اریب است. همچنین الگوریتمی برای پیاده‌سازی روش VN_n کلی و یک پیاده‌سازی مداری از VN_4 بیان می‌شود به‌طوری‌که VN_4 دارای نرخ خروجی ۴۶ درصد است. یک روش جدید تحت عنوان روش انتظار برای بهبود نرخ خروجی روش فون‌نویمان پیشنهاد می‌گردد به‌طوری‌که بر اساس روش جدید، VN_4 + waiting و VN_8 + waiting به ترتیب دارای نرخ‌های خروجی 46.9% و 62.5% هستند. در مقایسه با VN_2 که دارای نرخ خروجی 25% هست روش‌های جدید به ترتیب بهبود 1.88x و 2.5x را فراهم می‌کنند.

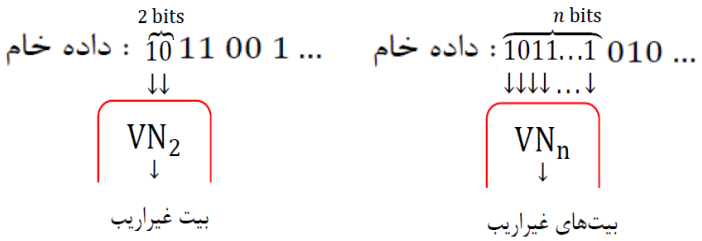
۴-۱-سابقه تحقیقاتی

روش فون‌نویمان برای اولین بار در سال ۱۹۵۱ پیشنهاد شد. در این روش با بریدن و حذف کردن برخی اطلاعات در رشته بیت ورودی دارای اریب می‌توان رشته بیت تصادفی غیر اریب تولید کرد. ولی این روش نرخ خروجی نزدیک به 25% طول دنباله ورودی را دارد. بعد از آن روش تکراری فون‌نویمان (IVN) پیشنهادشده است. با استفاده از تکرارهای نامتناهی روش IVN می‌توان به نرخ خروجی نزدیک به کران آنتروپی شانون دست یافت. تحت محدودیت منابع مداری، روش درخت دودویی غیر کامل برای دست‌یابی به خروجی‌هایی با کارایی بهینه پیشنهادشده است. به‌تازگی، روش $\mathrm{VN_n}$ پیشنهادشده که خروجی‌های کارآمدی را تولید می‌کند. نکته مهمی که وجود دارد این است که با افزایش مقدار n می‌توان تمام آنتروپی موجود در دنباله ورودی را با یک پساپردازش تئوری $\mathrm{VN_n}$ به دست آورد. اما باید توجه داشت که دست‌یابی به نرخ خروجی حاصل از مقادیر تئوری در دنیای واقعی و به ازای مقادیر واقعی یک مسئله سخت است.

همان‌طور که در شکل (۳) نشان داده‌شده است روش فون‌نویمان اصلی در هر مرتبه دو بیت را پردازش می‌کند درحالی‌که روش فون‌نویمان n بیتی در هر مرتبه n بیت را پردازش می‌کند. هر دو روش، بیت‌های غیر اریب را به‌عنوان خروجی تولید می‌کنند. به روش فون‌نویمان اصلی VN_2 نیز گفته می‌شود. نتایج و کارهای ارائه‌شده در این قسمت به‌صورت زیر هستند:

۱. روش VN_4 به‌صورت الگوریتمی و نیز بر اساس گیت‌های منطقی پیاده‌سازی شده است.
۲. یک روش جدید تحت عنوان روش انتظار نیز پیشنهادشده است.
۳. نتایج پیاده‌سازی نشان می‌دهند که VN_4 + waiting دارای نرخ خروجی 64.9% بوده و روش VN_8 + waiting دارای نرخ خروجی 62.5% است.

برای هر مقدار n موجود در VN_n، یک الگوریتم نگاشتی ارائه می‌گردد که از آن می‌توان در تولید جدول مراجعه استفاده نمود.



شکل (۳): فون‌نویمان اصلی و فون‌نویمان n بیتی

ادامه این بخش به‌صورت زیر بخش‌بندی شده است. در ابتدا، روش فون‌نویمان تکراری و فون‌نویمان n-بیتی بیان می‌شود. در ادامه، پیاده‌سازی روش فون‌نویمان n-بیتی شرح داده می‌شود. در پایان، نتایج حاصل از پیاده‌سازی بیان می‌گردد.

۴-۲-فون‌نویمان تکراری و فون‌نویمان n-بیتی

یک دنباله m بیتی را به‌صورت $x_1x_2 \dots x_m$ نشان خواهیم داد و فرض می‌کنیم که بیت‌های دنباله دارای توزیع مستقل و برابری هستند. اگر احتمال وقوع بیت‌های یک در دنباله بیتی برابر با p بوده و احتمال صفر بودن آن‌ها برابر با $q = 1 - p$ باشد در این صورت اریب (یا بایاس) دنباله بیتی به‌صورت زیر تعریف می‌شود.

$$e = \frac{|p-q|}{۲}, \quad ۰ \le e \le ۰.۵ \qquad (۱۴)$$

در ادامه، توضیح مختصر از روش فون‌نویمان تکراری بیان می‌گردد.

۴-۲-۱-فون‌نویمان تکراری (IVN)

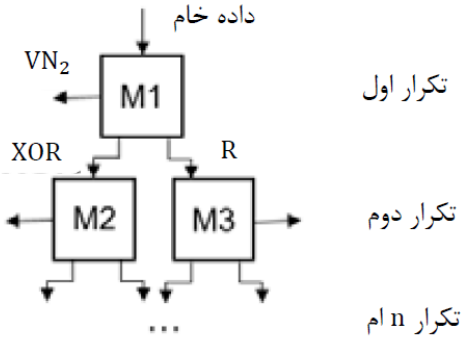
روش فون‌نویمان تکراری (IVN) به شرح زیر است: بعد از پیاده‌سازی VN_2، توابع R و XOR بر روی اطلاعات حذف‌شده (بیت‌های بریده و حذف‌شده در VN_2) اعمال می‌شوند و رشته بیت‌های جدیدی تولید می‌شوند. رشته بیت‌های جدید تولیدشده به‌عنوان ورودی VN_2 در مرحله جدید مورداستفاده قرار می‌گیرند. در شکل (۴)، توابع R و XOR با یک مثال ساده بیان‌شده‌اند و شکل (۵) ساختار فون‌نویمان تکراری را نشان می‌دهد. در روش IVN از سه تابع R، XOR و VN_2 استفاده می‌شود که به‌صورت زیر هستند:

- VN_2: اگر ورودی این تابع رشته بیت‌های 01 یا 10 باشد در این صورت خروجی آن بیت 0 یا 1 خواهد بود، در غیر این صورت هیچ خروجی تولید نخواهد شد.
- XOR: اگر ورودی این تابع رشته بیت‌های 00 یا 11 باشد در این صورت خروجی آن 0 خواهد بود، در غیر این صورت بیت 1 را تولید خواهد کرد.
- R: اگر ورودی این تابع رشته بیت 00 باشد در این صورت بیت 0 را تولید خواهد کرد، اگر ورودی آن 11 باشد در این صورت بیت 1 را تولید خواهد کرد، در غیر این‌صورت هیچ خروجی تولید نخواهد شد.

تابع VN_2 دارای نرخ خروجی برابر با $\%25 - e^2$ بوده و خروجی‌های آن بدون اریب هستند. تابع XOR دارای نرخ خروجی 50% با اریب $2e^2$ بوده و تابع R دارای نرخ خروجی $\%25 + e^2$ با اریب $\frac{2e}{1+4e^2}$ است.

داده خام	00	01	10	00	11	10	01	10	11	01	...
VN_2		0	1			1	0	1		0	...
XOR	0	1	1	0	0	1	1	1	0	1	...
R	0			0	1				1		...

شکل (۴): توابع استفاده‌شده در IVN با همراه یک مثال ساده



شکل (۵): ساختار روش IVN

خروجی‌های حاصل از R و XOR در روش IVN به‌عنوان ورودی‌های تابع VN_2 در مرحله جدید مورداستفاده قرار می‌گیرند. هراندازه تعداد تکرارهای روش IVN بیشتر باشد نرخ خروجی نیز بیشتر خواهد بود. ولی به علت تکرارهای بیشتر میزان اریب خروجی‌های تولیدشده نیز بیشتر خواهد بود و کارایی روش نیز کاهش خواهد یافت. برای مثال، اگر مقدار اریب داده خام برابر با 10% باشد بعد از دو مرتبه تکرار، مقدار اریب دنباله بیتی تولیدشده در ماژول M3 بیشتر از 30% خواهد بود.

۴-۲-۲-فون‌نویمان n بیتی

روش فون‌نویمان n-بیتی (VN_n) به شرح زیر است: در هر دنباله n بیتی 2^n جایگشت ممکن از بیت‌های آن وجود دارد. بر اساس روش VN_n، جایگشت‌های حاصل از دنباله بیتی به $n+1$ مجموعه S_k، $k=0,1,\dots,n$، تقسیم می‌شوند طوری که هر کدام از اعضای S_k دقیقاً دارای k بیت یک هست، k نشان‌دهنده تعداد بیت‌های یک موجود در اعضای آن مجموعه است. به دلیل اینکه احتمال وقوع هر کدام از اعضای مجموعه S_k برابر با $p^k\ q^{n-k}$ بوده، بنابراین بیت‌های اختصاص داده‌شده برای هر عضو مجموعه، فاقد اریب خواهند بود. تعداد بیت‌های اختصاص داده‌شده برای هر عضو از مجموعه S_k برابر با $\log_2 C_n^k$ بوده که در آن C_n^k برابر با تعداد اعضای مجموعه S_k است. دقت کنید که مقدار $\log_2 C_n^k$ بیشترین مقدار ایده‌آل هست و در عمل تعداد بیت‌های اختصاص داده‌شده برای اعضای مجموعه کمتر از مقدار $\log_2 C_n^k$ است. در جدول (۱) این مقادیر به‌صورت خلاصه‌وار ذکر شده‌اند. از لحاظ تئوری، نرخ خروجی روش VN_n برابر با مقدار زیر است:

$$\frac{\sum_{k=0}^{n} C_n^k \log_2\left(C_n^k\right)\ p^k\ \ q^{n-k}}{n} \qquad (۱۵)$$

برای یک نگاشت بیتی واقعی، بسط دودویی C_n^k به‌صورت زیر است:

$$C_n^k = a_{kn}2^n + a_{kn-1}2^{n-1} + \dots + a_{k0}2^0 = \sum_{j=0}^{n} a_{kj}2^j \qquad (۱۶)$$

در بسط دودویی بالا $a_{kn}=0$ و $a_{ki}\in\{0,1\}$ برای $i=0,1,\dots,n-1$ است. برای تمام $a_{kj}\neq 0$، $j=0,1,\dots,n$، بر اساس یک قاعده نگاشتی (در ادامه بیان خواهدشد) j بیت مختلف به هر کدام از اعضای مجموعه S_k اختصاص داده می‌شود. وقتی که $k\in\{0,n\}$ است

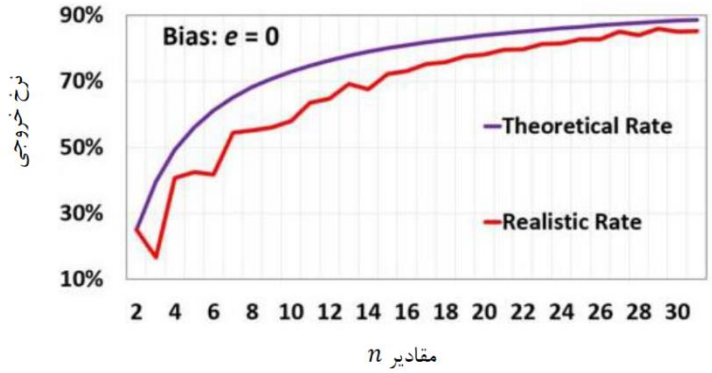
$C_n^k = 1$ خواهد بود و هیچ بیتی به اعضای مجموعه اختصاص داده نخواهد شد. مقدار واقعی نرخ خروجی روش VN_n برابر با مقدار زیر است.

$$\frac{\sum_{k=0}^{n}\sum_{j=0}^{n} a_{kj}\,2^{j}\,j\,p^{k}\,q^{n-k}}{n} \qquad (۱۷)$$

با افزایش مقدار n، با استفاده از روش VN_n می‌توان به نرخ خروجی نزدیک به کران آنتروپی شانون دست پیدا کرد. این بیان می‌کند که برای هر داده خام تولیدشده توسط یک TRNG می‌توان به خروجی‌هایی با بیشترین آنتروپی دست پیدا کرد. ولی هراندازه مقدار n زیاد باشد منبع مداری بیشتری نیز نیاز خواهد بود. در شکل (۶) نرخ خروجی تئوری و واقعی روش VN_n برای مقادیر مختلف n نشان داده‌شده است. برای $n = 4,8,16$ نرخ خروجی تئوری به ترتیب برابر با 49.2%، 68.2% و 81% است درحالی‌که نرخ خروجی واقعی به ترتیب برابر با 40.6%، 55.2% و 73.1% است.

جدول (۱): خلاصه‌ای از VN_n

S_k	تعداد اعضای S_k	احتمال وقوع هر عضو از S_k	تعداد بیت‌های اختصاص داده‌شده برای هر عضو (در حالت ایده‌آل)
S_0	1	q^n	0
S_1	n	$p\,q^{n-1}$	$\log_2 n$
⋮	⋮	⋮	⋮
S_k	C_n^k	$p^k\,q^{n-k}$	$\log_2 C_n^k$
⋮	⋮	⋮	⋮
S_n	1	p^n	0
مجموع	2^n	1	$\sum_{k=0}^{n} C_n^k \log_2(C_n^k)\;\; p^k\;\; q^{n-k}$



شکل (۶): نرخ خروجی روش VN_n به ازای مقادیر مختلف از n.

۴-۳-پیاده‌سازی و بهبود روش VN_n

۴-۳-۱-الگوریتم نگاشتی

یک الگوریتم نگاشتی برای پیاده‌سازی واقعی روش VN_n طراحی‌شده است که در الگوریتم ۱ به‌صورت شبه کد این الگوریتم نگاشتی (یا تابع نگاشتی) بیان‌شده است. از این الگوریتم می‌توان در تولید جدول-مراجعه در پیاده‌سازی ROM استفاده نمود.

در ادامه با مثالی الگوریتم نگاشتی را توضیح می‌دهیم. فرض کنید $n = 4$ و $k = 2$ باشد. در ابتدا مقدار C_4^2 محاسبه می‌شود که برابر با شش است. سپس این مقدار به یک بردار دودویی تبدیل می‌شود که به‌صورت $[0,1,1]$ است. این بردار دودویی به متغیر m اختصاص داده می‌شود و سپس به بردار $[0,2,4]$ به‌روزرسانی می‌شود. درایه صفر بردار حذف می‌شود و سپس m به بردار $[2,2,4,4,4,4]$ به‌روزرسانی می‌شود. حال تمام رشته بیت‌های از طول چهار در نظر گرفته می‌شوند و رشته بیت‌هایی که دارای $k = 2$ بیت غیر صفر هستند انتخاب

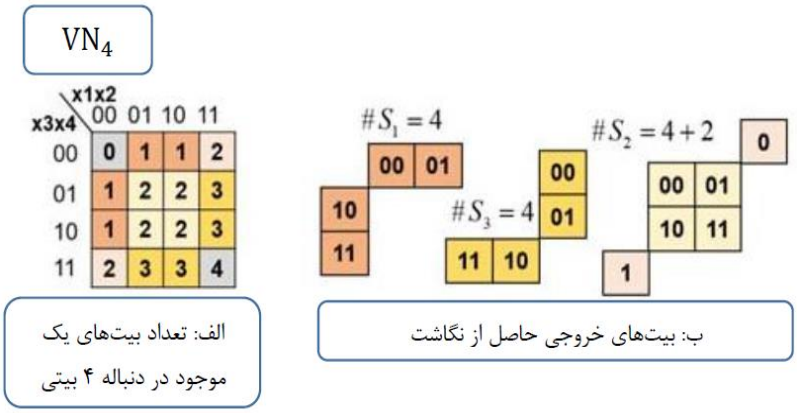
می‌شوند و سپس متغیرهای p_{value} و p_{bit} برابر با صفر قرار داده می‌شوند. در ادامه بر اساس الگوریتم ۱، متغیرهای p_{value} و p_{bit} به‌روزرسانی می‌شوند. برای $k = 2$ و $n = 4$، داریم $p_{value} \in \{1,2,3,4,5,6\}$ و $p_{bit} \in \{0,1,00,01,10,11\}$. فرآیند بالا برای تمام مقادیر ممکن k تکرار می‌شود و در نهایت اختصاص دادن بیت‌ها پایان می‌یابد. در بخش بعد، درباره پیاده‌سازی VN_4 بیشتر توضیح داده می‌شود.

الگوریتم (۱): تابع نگاشتی VN_n

1. map$[2^n] \leftarrow$ null; //e.g. $n = 4$
2. if $0 < k < n$ then // e.g. $k = 2$
   - 2.1. $m \leftarrow$ dec2binvec$(C_n^k - (C_n^k \%2))$; //$m = [0,1,1]$
   - 2.2. For any $m(i)$ in m
     - 2.2.1. $m(i) \leftarrow m(i) * 2^{(\text{length}(m)-1)}$; //$m = [0,2,4]$
   - 2.3. $m \leftarrow m(2{:}\,\text{length}(m))$; //$m = [2,4]$
   - 2.4. $m \leftarrow [m[1], m[1], m[2], m[2], \dots]$; // $m = [2,2,4,4,4,4]$
   - 2.5. $p_{value} \leftarrow 0$;
   - 2.6. $p_{bit} \leftarrow 0$;
   - 2.7. While $0 \leq j$ and $j \leq 2^n$ do
   - 2.8. If j contains k ones then
     - 2.8.1. map$(j) \leftarrow$ assign $\log_2(m(p_{value}))$ length of p_{bit};
     - 2.8.2. $p_{value} \leftarrow p_{value} + 1$;
     - 2.8.3. If $m(p_{value}) == m(p_{value} - 1)$ then
       - 2.8.3.1. $p_{bit} \leftarrow p_{bit} + 1$;
       - 2.8.3.2. Else $p_{bit} \leftarrow 0$;
3. End function;

۴-۳-۲-پیاده‌سازی VN_4

در این بخش یک پیاده‌سازی ساده ولی کارآمد از VN_4 ارائه می‌شود. در شکل (۷) قواعد نگاشتی استفاده‌شده در VN_4 نشان داده‌شده‌اند. در شکل (۷-الف) تعداد بیت‌های یک موجود در دنباله چهار بیتی نشان داده‌شده‌اند. دنباله‌های بیتی متشکل از چهار بیت به پنج مجموعه $S_0, S_1, \dots, S_4$ تقسیم می‌شوند. برای مثال $\#S_1 = 4$ بیان می‌کند که مجموعه S_1 که متشکل از رشته بیت‌های از طول چهار با یک بیت غیر صفر هست دارای چهار عضو هست که عبارت‌اند از $'1000`, '0100`, '0010`, '0001`$. احتمال وقوع هر عضو این مجموعه برابر با $p\,q^3$ است. همان‌طور که در شکل (۷-ب) نشان داده‌شده رشته بیت‌های $'10`, '01`, '00`$ و $'11`$ بر اساس الگوریتم ۱ به اعضای مجموعه S_1 اختصاص داده‌شده‌اند. مجموعه S_3 همانند مجموعه S_1 هست و دارای چهار عضو است. برای مجموعه S_2 چون $\#S_2 = 6$ بوده و $\log_2 C_4^2$ عدد صحیح نیست، اعضای مجموعه S_2 به یک مجموعه چهار عضوی و یک مجموعه دو عضو تجزیه‌شده‌اند. برای هر کدام از اعضای مجموعه چهار عضوی دو بیت اختصاص داده‌شده است و برای هر کدام از اعضای مجموعه دو عضوی یک بیت اختصاص داده‌شده است. بر این اساس روش VN_4، دنباله بیتی فاقد اریب را تولید می‌نماید.

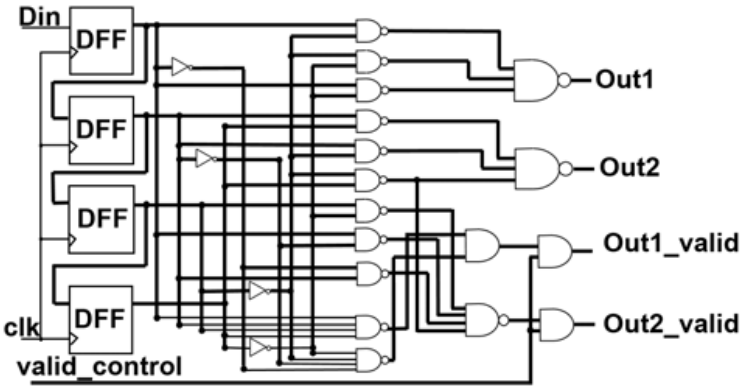


شکل (۷): نگاشت استفاده‌شده در VN_4

نرخ خروجی این روش برابر با مقدار زیر هست:

$$R_{VN_4} = \frac{۱۳}{۳۲} - \frac{۵}{۴}e^{۲} - \frac{۳}{۲}e^{۴} \qquad (۱۸)$$

درصورتی‌که دنباله ورودی فاقد اریب باشد در این صورت نرخ خروجی VN_4 برابر با 40.6% است درحالی‌که نرخ خروجی VN_2 برابر با 25% است. روش VN_4 در سطح مداری پیاده‌سازی شده و شکل (۸) پیاده‌سازی مداری آن را نشان می‌دهد.



شکل (۸): پیاده‌سازی مداری روش VN_4

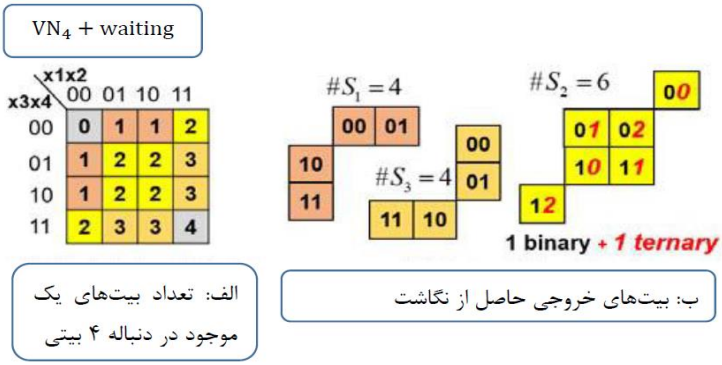
همان‌طور که در شکل (۸) مشاهده می‌کنید در این پیاده‌سازی از سه پورت ورودی (پورت‌های Din, clk و valid_control) و چهار پورت خروجی استفاده‌شده است. چهار پورت خروجی از دو پورت سیگنال داده out1 و out2 و دو پورت سیگنال معتبر out1_valid و out2_valid تشکیل‌شده‌اند. سیگنال‌های معتبر جهت مشخص کردن قابل‌استفاده بودن داده خروجی مورداستفاده قرار می‌گیرند. در جدول (۲) تحلیلی از میزان به‌کارگیری منبع این پیاده‌سازی برحسب معادل گیتی (GE) آورده شده است. همان‌طور که از جدول (۲) مشاهده می‌گردد با به‌کارگیری روش VN_4 مقدار GE برابر با 46.02 بوده که 23.3% کمتر از روش IVN3 است.

جدول (۲): تحلیلی از پیاده‌سازی VN_4 بر اساس معادل بیتی (GE).

گیت‌های استفاده‌شده	VN_2	VN_4	IVN3
INV (0.67)	1	4	6
AND2 (1.33)	1	3	12
XOR (2)	1	_	13
NAND2 (1)	_	9	_
NAND3 (1.33)	_	2	_
NAND4 (1.67)	_	3	_
DFF (5.67)	2	4	6
مجموع (GE)	15.34	46.02	60

۴-۳-۳-بهبود روش VN_n

در این بخش یک روش جدید برای تولید بیت‌های تصادفی تحت عنوان روش انتظار پیشنهاد می‌شود. با استفاده از این روش می‌توان نرخ خروجی واقعی بیت‌های تولیدشده توسط روش VN_n را به نرخ خروجی تئوری نزدیک نمود. روش جدید VN_n + waiting نامیده می‌شود. در شکل (۹) روش VN_4 + waiting نشان داده‌شده است. تفاوت این روش با روش VN_4 در نحوه اختصاص دادن بیت‌ها به اعضای مجموعه S_2 هست.



شکل (۹): نگاشت استفاده‌شده در روش $VN_4 + waiting$.

همان‌طور که گفتیم $\#S_2 = \log_2 C_4^2$ عدد صحیح نیست. در روش جدید یک بیت دودویی و یک بیت در مبنای سه $(`0', `1', `2')$ به اعضای مجموعه S_2 اختصاص داده می‌شود. از بیت‌های دودویی می‌توان به‌صورت مستقیم به‌عنوان خروجی استفاده کرد درحالی‌که از بیت در مبنای سه زمانی می‌توان به‌عنوان خروجی استفاده نمود که با بیت در مبنای سه، بعدی مطابقت داشته باشد. سپس بر اساس قاعده نگاشتی ذکرشده در شکل (۱۰) سه بیت به‌عنوان خروجی تولید می‌شود یا هیچ بیتی به‌عنوان خروجی تولید نمی‌شود.

t2 \ t1	2	1	0
2	000	001	010
1	100	101	110
0	011	111	-

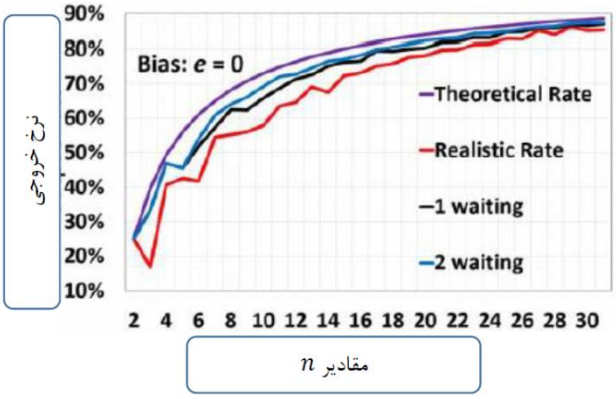
شکل (۱۰): نگاشتی از بیت‌های مبنای سه به بیت‌های دودویی.

نرخ خروجی روش $VN_4 + waiting$ برابر با مقدار زیر است:

$$R_{VN_4+waiting} = \frac{۱۵}{۳۲} - \frac{۷}{۴}\, e^{۲} - \frac{۱}{۲} e^{۴} \qquad (۱۹)$$

برای اریب $e = 0$ نرخ خروجی روش $VN_4 + waiting$ برابر 46.9% بوده که نزدیک به نرخ خروجی تئوری 49.2% است. با افزایش تعداد انتظارها می‌توان به رشته بیت‌هایی با آنتروپی نزدیک به کران تئوری دست پیدا نمود (منظور از تعداد انتظارها تعداد دفعاتی است که روش انتظار مورداستفاده قرار می‌گیرد).

شکل (۱۱) نرخ خروجی روش VN_n با یک و دو مرتبه انتظار را نشان می‌دهد. منحنی نرخ خروجی روش VN_n با یک مرتبه انتظار بین منحنی نرخ خروجی تئوری و واقعی قرار دارد. برای $n \leq 16$، این روش به‌صورت میانگین 6.1% بهبود را در مقایسه با منحنی نرخ خروجی واقعی فراهم می‌کند.



شکل (۱۱): نرخ خروجی روش $VN_n + waiting$.

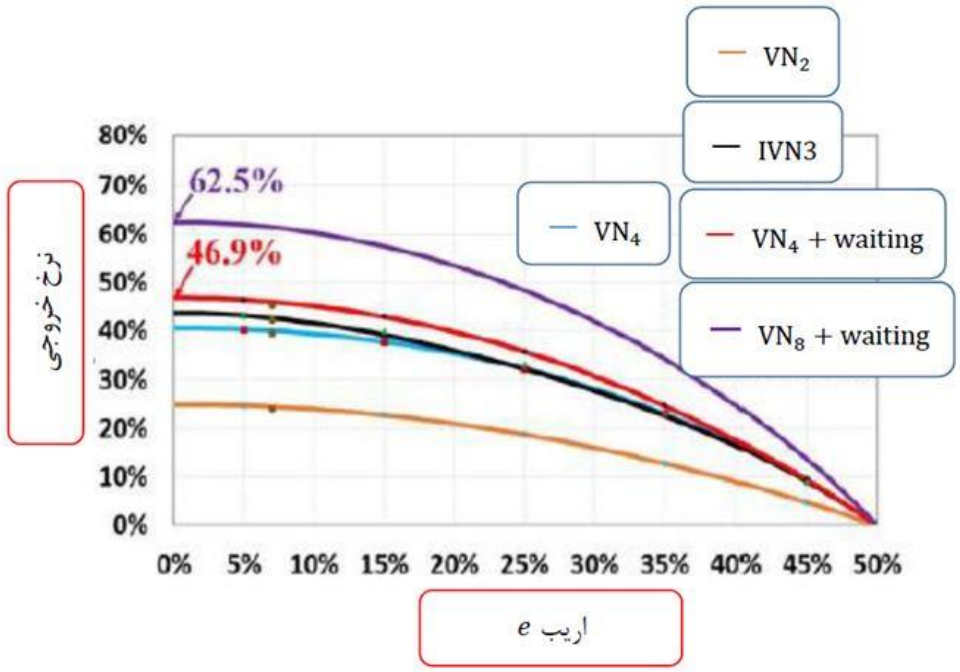
۴-۴-نتایج تجربی

از پنج دنباله بیتی شبه تصادفی دارای اریب و یک دنباله بیتی تصادفی واقعی دارای اریب به‌عنوان داده‌های ورودی آزمون استفاده‌شده است. طول هر کدام از دنباله‌های بیتی برابر با 10^6 است. در جدول (۳) نرخ خروجی و میانگین مقادیر p_{value} حاصل از آزمون‌های NIST داده‌شده است.

جدول (۳): نرخ خروجی و نتایج حاصل از آزمون NIST برای روش‌های مختلف تولید اعداد تصادفی.

منبع داده	اریب (e)	VN_2		IVN3		VN_4		VN_4 + waiting	
		نرخ خروجی	میانگین p_{value}	نرخ خروجی	میانگین p_{value}	نرخ خروجی	میانگین p_{value}	نرخ خروجی	میانگین p_{value}
واقعی	6.95%	24.09%	0.4794	42.30%	0.6820	39.49%	0.5976	45.35%	0.5595
شبه تصادفی	5.03%	24.68%	0.5382	43.24%	0.3164	40.26%	0.5346	46.45%	0.5731
	14.98%	22.82%	0.3709	39.95%	0.4092	37.80%	0.5407	42.98%	0.4162
	25.01%	18.76%	0.4550	33.27%	0.5357	32.22%	0.4693	35.75%	0.5047
	34.97%	12.77%	0.6731	23.38%	0.5608	23.09%	0.5189	24.71%	0.4483
	44.97%	4.77%	0.5229	9.21%	0.5515	9.20%	0.6004	9.43%	0.6256

نتایج حاصل از پیاده‌سازی روش‌های VN_2، VN_4، IVN3 (روش IVN با سه ماژول) و VN_4 + waiting با یکدیگر مقایسه شده و در شکل (۱۲) منحنی‌های نرخ خروجی این روش‌ها با اریب‌های مختلف نشان داده‌شده است. در شکل (۱۲)، منحنی‌ها نشان‌دهنده نرخ خروجی تئوری روش‌های ذکرشده بوده و نقاط مشخص‌شده بر روی منحنی‌ها نشان‌دهنده نتایج تجربی ذکرشده در جدول (۳) هستند. منحنی حاصل از نرخ خروجی تئوری روش VN_8 + waiting نیز در شکل (۱۲) رسم شده است.



شکل (۱۲): نرخ خروجی به ازای مقادیر مختلف اریب.

وقتی‌که اریب برابر با صفر باشد روش VN_8 + waiting دارای نرخ خروجی 62.5% است. روش VN_4 + waiting نسبت به روش IVN3 دارای نرخ خروجی بهتری است. وقتی مقدار اریب کمتر از 20% باشد روش IVN3 دارای کارایی بهتری نسبت به روش VN_4 است اما

باید توجه داشت وقتی‌که مقدار اریب افزایش پیدا می‌کند، به‌ویژه وقتی‌که اریب برابر با 25% باشد آنگاه نرخ خروجی روش‌های IVN3 و VN_4 برابر هستند.

۵-معرفی روش پساپردازش مبتنی بر ثبات انتقال با پس‌خورد خطی

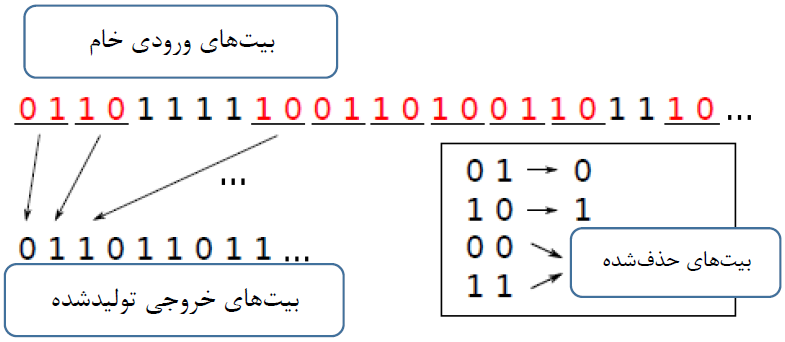
سیگنال نویزی از نوسانات کوانتومی سرچشمه می‌گیرد و غیرقابل‌پیش‌بینی است. ولی در بیشتر کاربردها نه‌تنها اعداد تصادفی باید غیرقابل‌پیش‌بینی باشند بلکه باید دارای توزیع یکنواخت نیز باشند. این بدین معناست که داده‌های خام ما که از دودویی نمودن سیگنال نویزی به‌دست‌آمده‌اند نمی‌توانند مستقیماً مورداستفاده قرار گیرند زیرا دارای توزیع غیریکنواخت هستند. بنابراین برای تشکیل یک طرح تولید اعداد تصادفی کامل نیاز به یک مرحله دیگر تحت عنوان مرحله پساپردازش داریم. در این قسمت، یک پیاده‌سازی از مولد اعداد تصادفی بر اساس ثبات انتقال با پس‌خورد خطی ($LFSR^{176}$) ارائه می‌گردد. بر اساس پیاده‌سازی‌های صورت گرفته، این روش نتایج خوبی را در برابر آزمون‌های تصادفی به دست آورده است.

۵-۱-اهمیت استفاده از LFSR در مقایسه با سایر روش‌ها

تولید (یا استخراج) اعداد تصادفی یک فرآیند ضروری برای تولید اعداد تصادفی باکیفیت بالا که دارای توزیع یکنواخت و ناهمبسته هستند، است. بخش اصلی تولید اعداد تصادفی یک الگوریتمی هست که به آن مولد اعداد تصادفی می‌گویند. مولد اعداد تصادفی یک رشته بیت با ویژگی‌های آماری ضعیف را به‌عنوان ورودی می‌گیرد و بیت‌های تصادفی که دارای توزیع یکنواخت هستند را به‌عنوان خروجی تولید می‌نماید.

روش فون‌نویمان یکی از مولدهای معروف برای تولید اعداد تصادفی هست. از روش فون‌نویمان برای تولید بیت‌های تصادفی با توزیع یکنواخت از یک دنباله بیتی مستقل ولی دارای اریب استفاده می‌شود (منظور از اریب این است که بیت‌های دنباله دارای بایاس می‌باشند؛ $P(0) \neq P(1) \neq \frac{1}{2}$). این روش در قسمت چهارم به‌طور کامل شرح داده‌شده اما برای کامل بودن این مقاله مروری و همچنین ارتباط آن با مطالب این قسمت، مجدد به‌طور مختصر توضیح داده می‌شود.

شکل (۱۳) روش فون‌نویمان برای تولید بیت‌های تصادفی را نشان می‌دهد. این روش دنباله‌ای از بیت‌های دارای اریب را به‌عنوان ورودی می‌گیرد و سپس آن را به زوج بیت‌هایی تقسیم می‌کند. تمام زوج‌هایی که به‌صورت 00 یا 11 هستند بریده و حذف می‌شوند. زوج‌هایی که به‌صورت 01 هستند به بیت 0 نگاشت می‌شوند و زوج‌هایی که به‌صورت 10 هستند به بیت 1 نگاشت می‌شوند. چون احتمال وقوع زوج‌هایی از فرم 01 برابر با احتمال وقوع زوج‌هایی از فرم 10 است دنباله بیتی تولیدشده توسط روش فون‌نویمان دارای یک توزیع یکنواخت است.



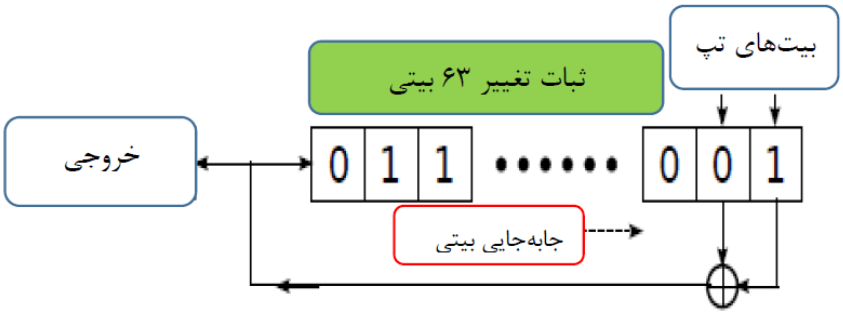
شکل (۱۳): روش فون‌نویمان برای تولید بیت‌های تصادفی.

نکته‌ای که وجود دارد این است که روش فون‌نویمان تنها برای ورودی‌هایی که بیت‌های آن مستقل و دارای اریب بوده مناسب است.

۵-۲-مولد اعداد تصادفی مبتنی بر LFSR

۵-۲-۱-ثبات انتقال با پس‌خورد خطی (LFSR)

یکی از روش‌های معروف برای تولید دنباله‌ای از بیت‌های طولانی با استفاده از محاسبات کم، استفاده نمودن از LFSR ها است. یک نمونه ساده از آن در شکل زیر آورده شده است.



شکل (۱۴): یک LFSR ۶۳ بیتی.

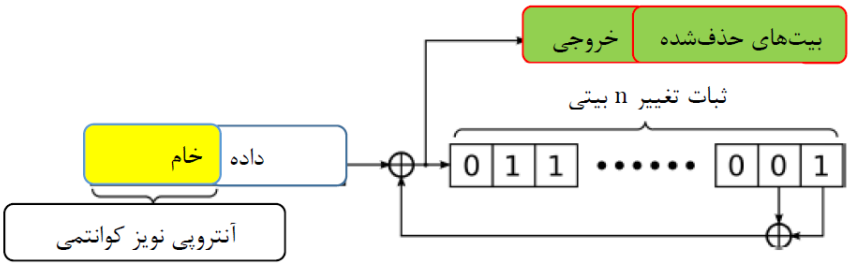
در شکل (۱۴) یک LFSR با ۶۳ بیت نمایش داده‌شده است. بخش اصلی این LFSR ثبات تغییر ۶۳ بیتی آن است. دو بیت سمت راست ثبات تغییر را بیت‌های تپ[۱۷۷] می‌نامند که عملیات XOR بر روی آن انجام می‌شود. با هر بار چرخه‌کلاک[۱۷۸] یک عملیات XOR اجرا می‌شود و یک بیت جدید عنوان خروجی تولید می‌شود. بیت‌های ثبات تغییر یک‌مرتبه به سمت راست جابه‌جا می‌شوند و بیت تولیدشده توسط عملیات XOR در خانه سمت چپ ثبات تغییر قرار می‌گیرد. با هر چرخه‌کلاک LFSR یک بیت را به‌عنوان خروجی تولید می‌کند و حالت داخلی ثبات تغییر نیز به‌روزرسانی می‌شود.

خروجی یک LFSR یک خروجی قطعی است بدین معنا که بعد از مدتی اگر دوباره LFSR را اجرا کنیم می‌توانیم خروجی را مجدداً تولید کنیم. این ویژگی به خاطر این است که تعداد متناهی حالت ممکن برای حالت‌های داخلی LFSR وجود دارد. با انتخاب مناسب موقعیت بیت‌های تپ یک LFSR می‌توان مدت‌زمان تولید مجدد را افزایش داد. برای یک LFSR با ثبات تغییر n-بیتی بیشترین مدت‌زمان تولید مجدد برابر با $2^n - 1$ بیت است. در LFSR ذکرشده در شکل (۱۴) دارای مدت‌زمان تولید مجدد $2^{63} - 1$ بیت بوده و همچنین دارای ساختار ساده‌ای هستند که ثبات تغییر 63 بیتی دارد. همچنین از دو بیت تپ برای تولید خروجی و به‌روزرسانی ثبات تغییر استفاده می‌کند. این LFSR کمترین تعداد ممکن از گیت‌های XOR را داشته و درعین‌حال بیشترین مدت‌زمان تولید مجدد را نیز دارد.

دنباله بیتی حاصل از خروجی LFSR هایی مشابه LFSR ذکرشده در شکل (۱۴) دارای ویژگی‌های آماری خوبی هستند. خروجی آن‌ها دارای توزیع یکنواخت است. بنابراین LFSR ها در برخی موارد برای تولید اعداد شبه تصادفی در ارتباطات مورداستفاده قرار می‌گیرند. همچنین LFSR ها با استفاده از مدارهای دیجیتال به‌راحتی قابل پیاده‌سازی هستند و سرعت بالایی نیز دارند. این ویژگی LFSRها باعث شده که با استفاده از آن‌ها مولد اعداد تصادفی با نرخ بسیار بالا طراحی شود.

۵-۲-۲-مولد اعداد تصادفی مبتنی بر LFSR

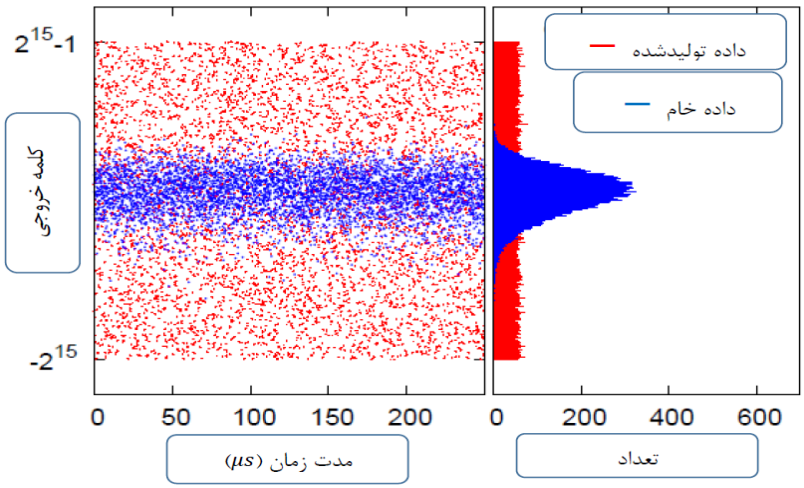
در شکل (۱۵) مولد اعداد تصادفی پیشنهادی نشان داده‌شده است. بخش اصلی این مولد یک LFSR با ثبات تغییر ۶۳ بیتی بوده که دارای حداکثر مدت‌زمان تولید مجدد است. داده‌خام به‌صورت سریال به مولد ارسال می‌شود. در هر چرخه‌کلاک یک عملیات XOR بر روی بیت‌های تپ اجرا می‌شود. بیت حاصل از این عملیات با بیت ورودی حاصل از داده‌خام XOR می‌شود و یک بیت جدید تولید می‌کند. بیت حاصل از این عملیات به‌عنوان خروجی تولید می‌گردد و همچنین به LFSR بازخورد می‌شود تا حالت داخلی LFSR به‌روزرسانی شود. از هر رشته بیت دلخواه از طول ۶۳ می‌توان به‌عنوان حالت داخلی آغازین مولد استفاده نمود و ۶۳ بیت ابتدایی خروجی‌های مولد باید حذف شوند تا مطمئن شویم که فرآیند تولید بیت‌های تصادفی برگشت‌ناپذیر است.



شکل (۱۵): مولد اعداد تصادفی مبتنی بر LFSR.

به دلیل اینکه، مولد اعداد تصادفی یک فرآیند قطعی بوده آنتروپی خروجی مولد اعداد تصادفی باید از آنتروپی داده‌خام ورودی آن کمتر باشد. داده‌های خام استفاده‌شده در مولد اعداد تصادفی دارای آنتروپی 14.1 بیت در هر نمونه ۱۶ بیتی هستند. برای اینکه مطمئن شویم آنتروپی خروجی‌های مولد اعداد تصادفی کمتر از کران آنتروپی شانون است هشت بیت هر خروجی ۱۶ بیتی را بریده و حذف می‌کنیم و تنها از ۸ بیت آن به‌عنوان خروجی نهایی استفاده می‌شود.

شکل (۱۶) تأثیر مولد اعداد تصادفی بر روی داده‌های خام ورودی را نشان می‌دهد. در این شکل توزیع حاصل از داده‌های خام ورودی و رشته بیت‌های تولیدشده توسط مولد در دامنه زمان (سمت چپ) و هیستوگرام (سمت راست) نمایش داده‌شده است. توزیع حاصل از داده‌های خام به‌صورت توزیع گاوسی است. بیت‌های تولیدشده توسط مولد به اعداد ۱۶ بیتی تبدیل‌شده‌اند و در شکل (۱۶) توزیع آن‌ها نشان داده‌شده است. همان‌طور که مشاهده می‌کنید اعداد تصادفی تولیدشده توسط مولد دارای توزیع یکنواخت هستند.

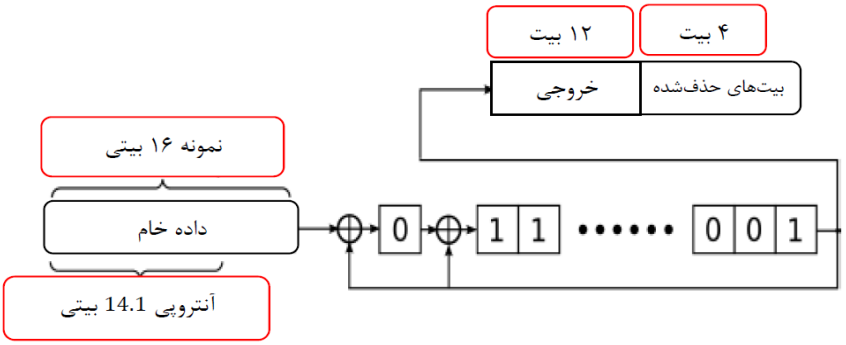


شکل (۱۶): تأثیر مولد اعداد تصادفی بر روی داده‌های خام ورودی

پیاده‌سازی مولد اعداد تصادفی ذکرشده در شکل (۱۵) برخی ویژگی‌های مثبت آن را روشن می‌کند. یکی از ویژگی‌های مثبت این مولد این است که دارای سرعت پیاده‌سازی بالایی است چون در هر چرخه کلاک فقط دو عملیات XOR اجرا می‌شود. برخلاف بسیاری از مولدهای اعداد تصادفی، این نوع از مولدها عامل محدودکننده‌ای برای نرخ تولید کلی نیستند. همچنین این نوع از مولدها بسیار فشرده هستند چون LFSR استفاده‌شده در آن‌ها دارای منابع محاسباتی بسیار کمی است. این نوع از مولدها را می‌توان در تلفن‌های همراه، کارت‌های هوشمند و برنامه‌های تلفن همراه که دارای توان محاسباتی محدود هستند استفاده نمود.

۵-۲-۳-مولدهای اعداد تصادفی با کارایی بالا

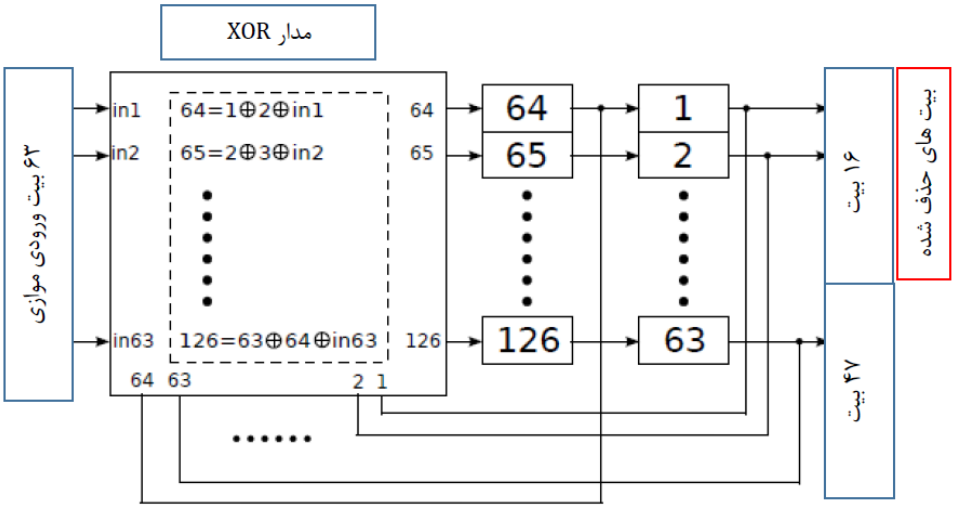
با استفاده از مولد اعداد تصادفی بیان‌شده در شکل (۱۵) می‌توان مولدهای اعداد تصادفی دیگری نیز طراحی کرد که حتی کارایی بهتری دارند. شکل (۱۷) یک مولد اعداد تصادفی مبتنی بر LFSR از نوع گالوا بوده که با اندکی تغییر در مولد ذکرشده در شکل (۱۵)، طراحی‌شده است.



شکل (۱۷): مولد اعداد تصادفی مبتنی بر LFSR از نوع گالوا.

یکی از مزیت‌های این مولد این است که عملیات XOR موجود در آن می‌توانند به‌صورت هم‌زمان انجام شوند و این باعث کاهش مدت‌زمان لازم برای پردازش ورودی‌های آن خواهد شد. توزیع حاصل از خروجی‌های این مولد مشابه با توزیع حاصل از خروجی‌های مولد ذکرشده در شکل (۱۵) است.

در شکل (۱۸) یک مولد اعداد تصادفی با بیت‌های ورودی به‌صورت موازی نشان داده‌شده است. این مولد قابلیت این را دارد که بیت‌های ورودی را به‌صورت موازی دریافت کند و بعد از پردازش آن‌ها بیت‌های خروجی را به‌صورت موازی تولید نماید. این قابلیت باعث افزایش سرعت پیاده‌سازی مولد اعداد تصادفی می‌شود. البته این ویژگی باعث می‌شود که تعداد گیت‌های XOR بیشتری در مدار پیاده‌سازی آن وجود داشته باشند درنتیجه برای پیاده‌سازی آن نیاز به حافظه بیشتری داریم.



شکل (۱۸): یک مولد اعداد تصادفی مبتنی بر LFSR با بیت‌های ورودی موازی.

۵-۳-آزمون اعداد تصادفی

تا این مرحله فرآیندی برای تولید اعداد تصادفی با استفاده از مولد اعداد تصادفی کوانتومی پیشنهاد شد. اما لازم است که کیفیت و میزان تصادفی بودن اعداد تولیدشده توسط مولد مورد تحلیل و بررسی قرار گیرد. تحقیقات زیادی در حوزه آزمون‌های اعداد تصادفی در حال انجام است. حقیقتی که وجود دارد این است که روشی که بتوان با استفاده از آن به‌صورت کامل و دقیق میزان تصادفی بودن اعداد را بررسی و تأیید کرد هنوز وجود ندارد. برای دنباله‌ای از اعداد تولیدشده از یک منبع نامشخص نمی‌توان گفت که آن‌ها اعداد تصادفی واقعی هستند چون همواره یک مولد اعداد تصادفی وجود دارد که می‌توان با استفاده از آن همان دنباله را تولید کرد. ولی با یک احتمالی می‌توان میزان تصادفی بودن اعداد تولیدشده توسط یک مولد را تعیین کرد. این کار با استفاده از آزمون‌های آماری امکان‌پذیر است. این آزمون‌ها ویژگی‌های آماری دنباله اعداد تولیدشده توسط یک مولد را بررسی می‌کنند و سپس آن‌ها را با مقادیر مورد انتظار دنباله اعداد

تصادفی واقعی مقایسه می‌کنند. این مقایسه معمولاً با استفاده از آزمون خی‌دو (یا آزمون توزیع نرمال) انجام می‌شود و میزان تصادفی بودن اعداد نیز بر اساس مقادیر p_{value} مشخص می‌شود.

خوشبختانه مجموعه‌ای از آزمون‌های تصادفی بودن جهت بررسی میزان تصادفی بودن اعداد وجود دارند. در این کار، برای بررسی میزان تصادفی بودن دنباله اعداد تولیدشده از مجموعه آزمون‌های آماری منتشرشده توسط NIST استفاده نموده‌ایم. گزارش‌های حاصل از آزمون‌های ذکرشده نشان می‌دهند که روش پیشنهادی مجموعه آزمون‌های آماری NIST را به‌خوبی پشت سر گذاشته است.

آزمون دیگری با استفاده از نرم‌افزارهای فشرده‌ساز انجام‌شده است. در این آزمون یک فایل دودویی از اندازه دو گیگابایت را با استفاده از مولد اعداد تصادفی پیشنهادشده تولید نموده‌ایم. سپس با استفاده از نرم‌افزارهای فشرده‌ساز مختلف به فشرده‌سازی آن پرداخته‌ایم که نتایج حاصل از آن در جدول (۴) آورده شده است. با توجه به اینکه دنباله بیتی تولیدشده توسط مولد یک دنباله تصادفی بوده، بنابراین الگوریتم‌های فشرده‌ساز موجود در نرم‌افزارهای فشرده‌ساز نتوانستند یک روش کارآمدی برای کدگذاری فایل پیدا کنند و به همین خاطر اندازه فایل فشرده تولیدشده تقریباً برابر با اندازه فایل اصلی است.

جدول (۴): بیت‌های تصادفی فشرده‌شده با استفاده از نرم‌افزارهای فشرده‌ساز.

نرم‌افزار فشرده‌ساز	اندازه فایل اصلی (بایت)	اندازه فایل فشرده‌شده	نرخ فشرده‌سازی
Lzma	2147438647	2176693974	1.0136
WinRAR	2147438647	2153045093	1.0026
Gzip	2147438647	2147831450	1.001
Bzip	2147438647	2156976807	1.0044

۶-مقایسه روش‌های معرفی‌شده

در ابتدا دو استخراج‌کننده Trevisan و تابع درهم‌ساز Toeplitz معرفی شدند. سرعت پیاده‌سازی برای استخراج‌کننده‌های Trevisan و تابع درهم‌ساز Toeplitz به ترتیب هفت‌دهم و ۴۴۱ کیلوبایت بر ثانیه است. نتایج حاصل از هر دو استخراج‌کننده، تمام آزمون‌های تصادفی استاندارد را پشت سر گذاشتند. در مقایسه با تابع درهم‌ساز Toeplitz، استخراج‌کننده Trevisan به مقدار اولیه‌ای با طول کوتاه‌تر نیاز دارد. بنابراین، استخراج‌کننده Trevisan در مواردی که بیت‌های مقادیر اولیه تصادفی محدود داشته باشند، دارای مزیت خاصی است. لازم به ذکر است که پیاده‌سازی اولیه استخراج‌کننده Trevisan به ما این اجازه را می‌دهد تا پیچیدگی و دشواری پیاده‌سازی استخراج‌کننده Trevisan را بهتر درک کنیم که این مسئله راه را برای پیاده‌سازی‌های آینده هموار می‌کند.

به دلیل سرعت پایین و پیچیدگی پیاده‌سازی سخت‌افزاری Trevisan و Toeplitz، روش فون‌نویمان معرفی گردید. ویژگی و جذابیت اصلی روش فون‌نویمان سادگی آن است. برای اجرای این روش به توان محاسباتی کمی نیاز است و با یک سخت‌افزار نسبتاً ساده می‌توان آن را پیاده‌سازی نمود. همچنین نیازی نیست که توزیع ورودی‌های این روش کاملاً شناخته‌شده باشند.

در روش فون‌نویمان تنها محدودیتی که بر روی دنباله ورودی وجود دارد این است که بیت‌های آن باید مستقل از هم باشند. اگر بین بیت‌های دنباله‌ای همبستگی وجود داشته باشد در این صورت این روش مؤثر نخواهد بود و نباید از آن استفاده نمود. همچنین با توجه به اینکه نرخ خروجی روش فون‌نویمان برابر با 25% بوده این روش یک روش کارآمد برای تولید بیت‌های تصادفی نیست.

به دلیل محدودیت ذکرشده برای روش فون‌نویمان، روش تولید اعداد تصادفی بر مبنای ثبات انتقال با پس‌خورد خطی شرح داده شد. مزیت‌های اصلی در تولید اعداد تصادفی به روش LFSR، سرعت پیاده‌سازی، سادگی پیاده‌سازی سخت‌افزاری و نداشتن محدودیت روش فون‌نویمان است. در این مورد این روش باید به این نکته مهم توجه شود که نرخ نمونه‌برداری مبدل ADC برابر با ۶۰ مگاهرتز بوده و هر نمونه به‌صورت عدد صحیح ۱۶ بیتی علامت‌دار است. در فرآیند تولید اعداد تصادفی، چهار بیت از هر نمونه ۱۶ بیتی حذف گردید و فقط از ۱۲ بیت آن به‌عنوان خروجی نهایی استفاده شد. ازلحاظ تئوری این کار سبب می‌شود تا نرخ تولید ۷۲۰ مگابیت در ثانیه را داشته باشیم. اما به خاطر محدودیت‌های موجود در انتقال داده‌ها، نرخ تولید در عمل برابر با ۴۸۰ مگابیت بر ثانیه است که البته از روش‌های

قبلی معرفی‌شده بسیار سریع‌تر است. نکته پایانی اینکه، با استفاده از ADC سریع‌تر و تقویت‌کننده‌هایی با پهنای باند وسیع‌تر، روش LFSR این قابلیت را دارد که اعداد تصادفی با نرخ تولید بیشتر از یک گیگابایت بر ثانیه را تولید نماید.

۷-نتیجه‌گیری

هر مولد اعداد تصادفی فیزیکی را می‌توان به چند بلوک مجزا، با وظایف و کارکردهای مشخص تقسیم کرد. دو بلوک منبع آنتروپی (بی‌نظمی) و مرحله پساپردازش مهم‌ترین این بلوک‌ها هستند. منبع آنتروپی شامل یک سامانه فیزیکی با یک یا چند کمیت فیزیکی تصادفی و تجهیزات اندازه‌گیری است که این متغیرهای تصادفی را می‌خواند. در مولدهای اعداد تصادفی دیجیتالی، معمولاً نیاز است که اندازه‌گیری‌های آنالوگ را به رشته‌های بیتی تبدیل کرد. حتی اگر کمیت اندازه‌گیری شده واقعاً تصادفی و عاری از هرگونه همبستگی و بایاس باشد، همواره به دلیل وجود ناکاملی و نقص در عملکرد قطعات، مقداری خطا در خروجی منبع آنتروپی وجود خواهد داشت. بلوک پساپردازش، این بیت‌ها را می‌گیرد و یک دنباله کوتاه‌تر و بدون همبستگی را از آن‌ها می‌سازد. درواقع، مولدهای اعداد تصادفی استاندارد برای تولید رشته‌های تصادفی با توزیع یکنواخت طراحی‌شده‌اند. مرحله پساپردازش موجود در این مولدها، دنباله بیتی خام را به رشته‌ای از بیت‌های تصادفی تبدیل نموده که توزیع حاصل از آن‌ها تا حد ممکن به توزیع یکنواخت نزدیک است. در این مقاله، در ابتدا روش‌های ارزیابی خروجی مولد اعداد تصادفی، شرح داده‌شده و سپس روش‌های پساپردازش متداول Toeplitz و Trevisan همراه با نحوه پیاده‌سازی آن‌ها آورده شده است. همچنین، روش پساپردازشی فون‌نویمان با توان عملیاتی بالا برای تولید اعداد تصادفی معرفی‌شده است. در پایان، یک پیاده‌سازی از مولد اعداد تصادفی بر اساس ثبات انتقال با پس‌خورد خطی (LFSR) شرح داده می‌شود.

مراجع

[1] W. Schindler and W. Killmann, “Evaluation criteria for true (physical) random number generators used in cryptographic applications,” in *Cryptographic Hardware and Embedded Systems (CHES)*, 2002, pp. 431-449, doi: 10.1007/3-540-36400-5_31.

[2] S. Pironio and S. Massar, “Security of practical private randomness generation,” *Physical Review A*, vol. 87, no. 1, pp. 1-12, Jan. 2013, doi: 10.1103/PhysRevA.87.012336.

[3] V. Neumann, “Various techniques used in connection with random digits,” in *Notes by GE Forsythe*, 1951, pp. 36-38.

[4] P. Elias, “The efficient construction of an unbiased random sequence,” *The Annals of Mathematical Statistics*, vol. 43, no. 3, pp. 865-870, Jun. 1972, doi: 10.1214/aoms/1177692552.

[5] Y. Peres, “Iterating von Neumann's procedure for extracting random bits,” *The Annals of Statistics*, vol. 20, no. 1, pp. 590-597, Mar. 1992, doi: 10.1214/aos/1176348543.

[6] D. Zuckerman, “Simulating BPP using a general weak random source,” *Algorithmica*, vol. 16, no. 14, pp. 367-391, Oct. 1996, doi: 10.1007/BF01940870.

[7] Y. Dodis, S.J. Ong, M. Prabhakaran and A. Sahai, “On the impossibility of cryptography with imperfect randomness,” in *45th Annual IEEE Symposium on Foundations of Computer Science*, 2004, pp. 196-205, doi: 10.1109/FOCS.2004.44.

[8] S. Vadhan, “The unified theory of pseudorandomness: Guest column,” *ACM SIGACT News*, vol. 38, no. 3, pp. 39-54, Sep. 2007, doi: 10.1145/1324215.1324225.

[9] R. Shaltiel, “An Introduction to Randomness Extractors,” in *Automata, languages and programming (ICALP)*, 2011, pp. 21-41, doi: 10.1007/978-3-642-22012-8_2.

[10] N. Nisan, “Extracting randomness: how and why. A survey,” in *Proceedings of Computational Complexity (IEEE Computer Society Press)*, 1996, pp. 44-58, doi: 10.1109/CCC.1996.507667.

[11] N. Nisan and A. Ta-Shma, “Extracting Randomness: A Survey and New Constructions,” *Journal of Computer and System Sciences*, vol. 58, no. 1, pp. 148-173, Feb. 1999, doi: 10.1006/jcss.1997.1546.

[12] R. Shaltiel, “Recent developments in explicit constructions of extractors,” in *Current Trends in Theoretical Computer Science*, 2004, pp. 189-228, doi: 10.1142/9789812562494_0013.

[13] X. Ma, F. Xu, H. Xu, X. Tan, B. Qi and H. Lo, “Postprocessing for quantum random-number generators: Entropy evaluation and randomness extraction,” *Physical Review*, vol. 87, no. 6, pp. 1-10, Jun. 2013, doi: 10.1103/physreva.87.062327.

[14] B. Chor and O. Goldreich, "Unbiased bits from sources of weak randomness and probabilistic communication complexity," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 230-261, Apr. 1988, doi: 10.1137/0217015.

[15] L. Trevisan and S. Vadhan, "Extracting randomness from samplable distributions," in *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, 2000, pp. 32-42, doi: 10.1109/SFCS.2000.892063.

[16] A. Gabizon, R. Raz and R. Shaltiel, "Deterministic extractors for bit-fixing sources by obtaining an independent seed," *SIAM Journal on Computing*, vol. 36, no. 4, pp. 1072-1064, Jan. 2006, doi: 10.1137/s0097539705447049.

[17] J. Kamp and D. Zuckerman, "Deterministic extractors for bit-fixing sources and exposure-resilient cryptography," *SIAM Journal on Computing*, vol. 36, no. 5, pp. 1231-1247, Jan. 2007, doi: 10.1137/s0097539705446846.

[18] J. Bourgain, "On the construction of affine extractors," *GAFA Geometric and Functional Analysis*, vol. 17, no. 1, pp. 33-57, Feb. 2007, doi: 10.1007/s00039-007-0593-z.

[19] A. Gabizon and R. Raz, "Deterministic extractors for affine sources over large fields," *Combinatorica*, vol. 28, no. 4, pp. 415-440, Jul. 2008, doi: 10.1007/s00493-008-2259-3.

[20] Z. Dvir, "Extractors for varieties," *Computational Complexity*, vol. 21, no. 4, pp. 515-572, Oct. 2011, doi: 10.1007/s00037-011-0023-3.

[21] M. Blum, "Independent unbiased coin flips from a correlated biased source: a finite state Markov chain," *Combinatorica*, vol. 6, no. 2, pp. 97-108, Jun. 1986, doi: 10.1007/BF02579167.

[22] H. Zhou and J. Bruck, "Efficient generation of random bits from finite state Markov chains," *IEEE Transactions on Information Theory*, vol. 58, no. 4, pp. 2490-2506, Dec. 2011, doi: 10.1109/TIT.2011.2175698.

[23] M. Santha and U.V. Vazirani, "Generating quasi- random sequences from semi-random sources," *Journal of Computer and System Sciences*, vol. 33, no. 1, pp. 75-87, Aug. 1986, doi: 10.1016/0022-0000(86)90044-9.

[24] A.E. Andreev, A.E.F. Clementi, J.D.P. Rolim and L. Trevisan, "Weak random sources, hitting sets, and BPP simulations," *SIAM Journal on Computing*, vol. 28, no. 6, pp. 2103-2116, Jan. 1999, doi: 10.1137/s0097539797325636.

[25] U.V. Vazirani and V.V. Vazirani, "Random polynomial time is equal to slightly-random polynomial time," in *26th Annual Symposium on Foundations of Computer Science*, 1985, pp. 417-428, doi: 10.1109/SFCS.1985.45.

[26] P. Austrin, K.M. Chung, M. Mahmoody, R. Pass and K. Seth, "On the impossibility of cryptography with tamperable randomness," in *Advances in Cryptology (CRYPTO)*, 2014, pp. 462-479, doi: 10.1007/978-3-662-44371-2_26.

[27] Y. Dodis, A. Elbaz, R. Oliveira and R. Raz, "Improved randomness extraction from two independent sources," in *Proceedings RANDOM*, 2004, pp. 334-344, doi: 10.1007/978-3-540-27821-4_30.

[28] Y. Dodis and J. Spencer, "On the (non)universality of the one-time pad," in *Foundations of Computer Science*, 2002, pp. 376-385, doi: 10.1109/SFCS.2002.1181962.

[29] J.L. McInnes and B. Pinkas, "On the impossibility of private key cryptography with weakly random keys," in *Advances in Cryptology (CRYPTO)*, 1991, pp. 421-435, doi: 10.1007/3-540-38424-3_31.

[30] U.V. Vazirani, "Strong communication complexity or generating quasi–random sequences from two communicating semi–random sources," *Combinatorica*, vol. 7, no. 4, pp. 375-392, Dec. 1987, doi: 10.1007/BF02579325.

[31] U. Vazirani, "Efficiency considerations in using semi-random sources," in *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, 1987, pp. 160-168, doi: 10.1145/28395.28413.

[32] B. Barak, R. Impagliazzo and A. Wigderson, "Extracting randomness using few independent sources," *SIAM Journal on Computing*, vol. 36, no. 4, pp. 1095-1118, Jan. 2006, doi: 10.1137/S0097539705447141.

[33] J. Bourgain, "More on the sum–product phenomenon in prime fields and its applications," *International Journal of Number Theory*, vol. 1, no. 1, pp. 1-32, Mar. 2005, doi: 10.1142/s1793042105000108.

[34] R. Raz, "Extractors with weak random seeds," in *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, 2005, pp. 11-20, doi: 10.1145/1060590.1060593.

[35] R. Shaltiel, "How to get more mileage from randomness extractors," *Random Structures and Algorithms*, vol. 33, no. 2, pp. 157-186, Sep. 2008, doi: 10.1002/rsa.20207.
[36] N. Nisan and D. Zuckerman, "Randomness is Linear in Space," *Journal of Computer and System Sciences*, vol. 52, no. 1, pp. 43-52, Feb. 1996, doi: 10.1006/jcss.1996.0004.
[37] N. Alon and J.H. Spencer, "The Basic Method," in *The Probabilistic Method*, Wiley, 2004, pp. 1-12, doi: 10.1002/0471722154.ch1.
[38] J. Radhakrishnan and A. Ta-Shma, "Bounds for Dispersers, Extractors, and Depth-Two Superconcentrators," *SIAM Journal on Discrete Mathematics*, vol. 13, no. 1, pp. 2-24, Jan. 2000, doi: 10.1137/s0895480197329508.
[39] C.J. Lu, O. Reingold, S. Vadhan and A. Wigderson, "Extractors," in *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, 2003, pp. 602-611, doi: 10.1145/780542.780630.
[40] A. Ta-Shma, "On extracting randomness from weak random sources (extended abstract)," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 276-285, doi: 10.1145/237814.237877.
[41] O. Goldreich and A. Wigderson, "Derandomization That Is Rarely Wrong from Short Advice That Is Typically Good," in *Proceedings of RANDOM*, 2002, pp. 209-223, doi: 10.1007/3-540-45726-7_17.
[42] A. Ben-Aroya and A. Ta-Shma, "Better short-seed quantum-proof extractors," *Theoretical Computer Science*, vol. 419, pp. 17-25, Feb. 2012, doi: 10.1016/j.tcs.2011.11.036.
[43] L. Trevisan, "Extractors and pseudorandom generators," *Journal of the ACM*, vol. 48, no. 14, pp. 860-879, Jul. 2001, doi: 10.1145/502090.502099.
[44] A. De, C. Portmann, T. Vidick and R. Renner, "Trevisan's Extractor in the Presence of Quantum Side Information," *SIAM Journal on Computing*, vol. 41, no. 4, pp. 915-940, Jan 2012, doi: 10.1137/100813683
[45] A. De and T. Vidick, "Near-optimal extractors against quantum storage," in *Proceedings of the Forty second ACM Symposium on Theory of Computing*, 2010, pp. 161-170, doi: 10.1145/1806689.1806713.
[46] A. Ta-Shma, "Short Seed Extractors against Quantum Storage," *SIAM Journal on Computing*, vol. 40, no. 3, pp. 664-677, Jan. 2011, doi: 10.1137/09076787x.
[47] W. Mauerer, C. Portmann and V.B. Scholz, "A modular framework for randomness extraction based on Trevisan's construction," *arXiv*, Dec. 2012, pp. 1-21, doi: 10.48550/arXiv.1212.0520.
[48] N. Nisan and A. Wigderson, "Hardness vs randomness," *Journal of Computer and System Sciences*, vol. 49, no. 2, pp. 149-167, Oct. 1994, doi: 10.1016/s0022-0000(05)80043-1.
[49] J. HAstad, R. Impagliazzo, L. A. Levin and M. Luby, "A Pseudorandom Generator from any One-way Function," *SIAM Journal on Computing*, vol. 28, no. 4, pp. 1364-1396, Jan. 1999, doi: 10.1137/s0097539793244708.
[50] R. Impagliazzo, L.A. Levin and M. Luby, "Pseudorandom generation from one-way functions," in *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, 1989, pp. 12-24, doi: 10.1145/73007.73009.
[51] J.L. Carter and M.N. Wegman, "Universal classes of hash functions," *Journal of Computer and System Sciences*, vol. 18, no. 2, pp. 143-154, Apr. 1979, doi: 10.1016/0022-0000(79)90044-8.
[52] M.N. Wegman and J.L. Carter, "New hash functions and their use in authentication and set equality," *Journal of Computer and System Sciences*, vol. 22, no. 3, pp. 265-279, Jun. 1981, doi: 10.1016/0022-0000(81)90033-7.
[53] M. Tomamichel, C. Schaffner, A. Smith and R. Renner, "Leftover Hashing Against Quantum Side Information," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5524-5535, Aug. 2011, doi: 10.1109/tit.2011.2158473.
[54] R. Konig and R. Renner, "Sampling of Min-Entropy Relative to Quantum Knowledge," *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4760-4787, Jul. 2011, doi: 10.1109/tit.2011.2146730.
[55] R.T. Konig and B.M. Terhal, "The Bounded-Storage Model in the Presence of a Quantum Adversary," *IEEE Transactions on Information Theory*, vol. 54, no. 2, pp. 749-762, Feb. 2008, doi: 10.1109/tit.2007.913245.
[56] C.H. Bennett, G. Brassard, C. Crepeau and U.M. Maurer, "Generalized privacy amplification," *IEEE Transactions on Information Theory*, vol. 41, no. 6, pp. 1915-1923, Nov. 1995, doi: 10.1109/18.476316.

[57] C.H. Bennett, G. Brassard and J.M. Robert, "Privacy Amplification by Public Discussion," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 210-229, Apr. 1988, doi: 10.1137/0217014.
[58] R. Renner and R. König, "Universally Composable Privacy Amplification Against Quantum Adversaries," in *Lecture Notes in Computer Science in Theory of Cryptography*, 2005, pp. 407-425, doi: 10.1007/978-3-540-30576-7_22.
[59] R. Konig, U. Maurer and R. Renner, "On the Power of Quantum Memory," *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2391-2401, Jul. 2005, doi: 10.1109/tit.2005.850087.
[60] C. Portmann and R. Renner, "Security in quantum cryptography," *Reviews of Modern Physics*, vol. 94, no. 2, Jun. 2022, doi: 10.1103/revmodphys.94.025008.
[61] A. Navarrete, V. Zapatero and M. Curty, "Quantum Key Distribution Protocols," *Photonic Quantum Technologies*, pp. 85–103, May 2023, doi: 10.1002/9783527837427.ch5.
[62] B. Barak, R. Shaltiel and E. Tromer, "True Random Number Generators Secure in a Changing Environment," in *Cryptographic Hardware and Embedded Systems (CHES)*, 2003, pp. 166-180, doi: 10.1007/978-3-540-45238-6_14.
[63] M. Skorski, "True Random Number Generators Secure in a Changing Environment: Improved Security Bounds," in *Theory and Practice of Computer Science*, 2015, pp. 590-602, doi: 10.1007/978-3-662-46078-8_49.
[64] F.N. Bostancı, A. Olgun, L. Orosa, "DR-STRaNGe: end-to-end system design for DRAM-based true random number generators," In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 2022, pp. 1141-1155, doi: 10.1109/HPCA53966.2022.00087.
[65] M.M. Fazili, M.F. Shah, S.F. Naz and A.P. Shah, "Next generation QCA technology based true random number generator for cryptographic applications," *Microelectronics Journal*, vol. 126, p. 495-502, Aug. 2022, doi: 10.1016/j.mejo.2022.105502.
[66] H. Krawczyk, "LFSR-based Hashing and Authentication," in *Advances in Cryptology (CRYPTO)*, 1994, pp. 129-139, doi: 10.1007/3-540-48658-5_15.
[67] Y. Mansour, N. Nisan and P. Tiwari, "The computational complexity of universal hashing," *Theoretical Computer Science*, vol. 107, no. 1, pp. 121-133, Jan. 1993, doi: 10.1016/0304-3975(93)90257-t.
[68] D. Frauchiger, R. Renner and M. Troyer, "True randomness from realistic quantum devices," *arXiv*, Nov. 2013, pp. 1-12, doi: 10.48550/arXiv.1311.4547.
[69] O. Guillan-Lorenzo, M. Troncoso-Costas and D. Alvarez-Outarelo, "Optical quantum random number generators: a comparative study," *Optical and Quantum Electronics*, vol. 55, no. 2, p. 170-185, Feb. 2023, doi: 10.1007/s11082-022-04396-y.
[70] A. Gholipour and S. Mirzakuchaki, "A Pseudorandom Number Generator with KECCAK Hash Function," *International Journal of Computer and Electrical Engineering*,2011, pp. 896–899, doi: 10.7763/ijcee.2011.v3.439.
[71] H. Seo, J. Choi, H. Kim, T. Park and H. Kim, "Pseudo random number generator and Hash function for embedded microprocessors," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, 2014, pp. 37-40, doi: 10.1109/wf-iot.2014.6803113.

زیرنویس‌ها

[1] Random uniform string
[2] Postprocessing
[3] Raw bit sequence
[4] RNGs
[5] Biases and correlations
[6] Imperfection
[7] High entropy
[8] Cryptographic hash function
[9] Von Neumann
[10] Debiasing

[11] Weak sources
[12] Randomized algorithms
[13] Bit commitment
[14] Zero knowledge proof
[15] Secret sharing
[16] Acceptably
[17] Support
[18] Ideal RNG
[19] Deviation
[20] Distinguishability
[21] Composable
[22] Composability
[23] Classical cryptographic protocols
[24] Partially
[25] Almost perfect
[26] Two-universal
[27] Dispersers
[28] Condensers
[29] Error-correcting codes
[30] Samplers
[31] Well characterized
[32] Markov process
[33] Deterministic extractors
[34] General deterministic
[35] Reasonable sources
[36] Practical deterministic
[37] Bit-fixing
[38] Affine sources
[39] Unknown algebraic variety
[40] Shannon entropy
[41] Markov chain
[42] Santha-Vazirani
[43] Signature schemes
[44] Multiple source extractors
[45] Concrete input distributions
[46] bitwise AND
[47] Seed
[48] Catalyst
[49] Contradiction
[50] Explicit
[51] Exhaustive enumeration
[52] Majority voting
[53] Unpredictability
[54] Quantum attackers
[55] Trevisan extractor
[56] Nisan-Widgerson
[57] Truth table
[58] Leftover hash lemma
[59] Two-universal hashing
[60] Conservative
[61] Side information
[62] Technical noise
[63] Toeplitz
[64] Secure hashing algorithm
[65] Advanced encryption standard
[66] Pseudorandom-number generators
[67] Computational complexities
[68] Deterministic algorithm

[69] Bugs
[70] Imperfection
[71] Physical RNGs
[72] Probabilistic nature
[73] Microprocessors
[74] Quantum signals
[75] True randomness
[76] Classical noises
[77] Adversary
[78] Partial information
[79] Trusted device
[80] Calibrate
[81] Deterministic
[82] Distilling
[83] Randomness extraction
[84] Extractors
[85] Raw data
[86] Short random seed
[87] Min-entropy
[88] Quantify
[89] Bitwise XOR operation
[90] Least-significant-bits
[91] Hashing functions
[92]Sophisticated
[93]Shannon entropy
[94]Computational assumptions
[95]Polylogarithmic
[96]Privacy amplification
[97]Quantum key distribution (QKD)
[98]Net randomness
[99] Quantification of randomness
[100] Gaps
[101]Uniform distribution
[102]Prototypical implementation
[103]Real-life QRNGs
[104]Major computational step
[105]Reasonable assumptions
[106]Characterize
[107]Setup
[108]Measurements
[109] Certain measurement
[110] Quantum signal
[111]Inevitably
[112]Background detections
[113]Electronic noises
[114]Manipulated
[115] Quantum state
[116]Truly random numbers
[117] Emissions
[118]Spontaneous
[119]Threshold level
[120]Self-heterodyne
[121]Delayed
[122] Quadrature amplitude
[123]Vacuum state
[124]Homodyne detector
[125]Hash extractor
[126]Detailed procedures

[127]Vacuum fluctuation
[128]Phase fluctuations
[129] Total signal
[130]Gaussian distribution
[131]Analog to digital conversion
[132]Quantum-to-classical ratio
[133]Eight-bit ADC
[134]Digitalized output
[135]Evenly
[136]Voltage axis
[137] Quantitatively separating
[138] Contributions
[139]Margin
[140]Upper bound of entropy
[141]Photo detector
[142]Perfect photon-number resolving detector
[143]Detection time window
[144] One-bit extractor
[145]Combinatorial design
[146]Error correcting code
[147]Concatenating
[148] Reed-Solomon code
[149]Hadamard code
[150]Refined version
[151]Nisan-Wigderson
[152]Major computational
[153] One-bit extractor
[154]Error-correction-based
[155] Ratio of min-entropy
[156]Square of the smallest power
[157]Substring
[158]Privacy amplification
[159]Subtle
[160]Net randomness
[161]Subsequent applications
[162] Compromising
[163]Privacy of the seed
[164]Standard QKD postprocessing
[165]Raw key data
[166]Finite-size effect
[167]Efficiency
[168] Extracted random-bit string
[169]Bits per sample
[170] Normalized correlation
[171] Vicinity
[172]Independent and identically distributed
[173]Practical detector
[174]Statistical fluctuations
[175]Low residual values
[176] Linear feedback shift register
[177] Tap bits
[178] Clock Cycle