نشر ریاضی، سال ۱۴، شمارهٔ ۱، صص. ۱۰ـ۱۶

اول بودنْ یک مسألهٔ P است

حسن حقیقی*

در روز یکشنبه چهارم اوت ۲۰۰۲، مانیندرا اگراوال[1]، استاد مؤسسهٔ تکنولوژی کانپور هند، به همراه دو دانشجوی خود، ساکسنا[2] و کایال[3]، الگوریتمی برای تعیین اول‌بودن یک عدد طبیعی ارائه دادند که نه تنها تعداد عملیات لازم برای اجرای آن تابعی چندجمله‌ای از طول عدد در یک دستگاه شمار است بلکه می‌تواند اول‌بودن یا نبودن هر عدد طبیعی را به‌طور قطعی تعیین کند. به این ترتیب، مسألهٔ قدیمیِ

تعیین اول‌بودن یک عدد طبیعی یک مسألهٔ P است یا NP

که در حوزهٔ نظریهٔ الگوریتمی اعداد و نظریهٔ پیچیدگی محاسبه مطرح بود، حل شد.

ایشان ابتدا راه‌حل خود را، که در مقاله‌ای ۹ صفحه‌ای تنظیم شده بود، برای ۱۵ نفر از متخصصان تراز اول این دو رشته ازجمله کارل پومرانس[4]، هندریک لنسترا[5]، دانیل برنستاین[6]، و چند هموطن هندی خود فرستادند و نظر آنها را در مورد درستی آن جویا شدند. در همان روز متخصصان هندی موفقیت آنها را تبریک گفتند. روز بعد پومرانس با جملهٔ «این بهترین نتیجه‌ای است که در طی ۱۰ سال گذشته به آن برخورده‌ام» مهر تأییدی بر درستیِ کار آنها زد و در عصر روز دوشنبه پنجم اوت، سمیناری برای توضیح دستاورد آنها در دانشگاه خود ترتیب داد و از سارا رابینسن، یکی از نویسندگان بخش علمی روزنامهٔ نیویورک تایمز نیز برای شرکت در این سمینار دعوت به‌عمل آورد. به‌دنبال این تأیید، اگراوال مقالهٔ مذکور را از طریق وب‌گاه خود [۳] در دسترس همگان قرار داد و خانم رابینسن نیز، بر اساس آنچه در سمینار پومرانس عنوان شده بود، خبر حل مسألهٔ فوق را در روز هشتم اوت، از طریق روزنامه نیویورک تایمز، به اطلاع عموم رساند و به دنبال آن سیل‌نامه‌های الکترونیکی حاوی پیامهای تبریک به‌سوی حل‌کنندگان مسأله سرازیر شد. هندریک لنسترا با رفع چند اشتباه جزئی در مقاله، درستی اثبات را تأیید کرد و در هشتم اوت، دانیل برنستاین نیز با مختصر تغییری در استدلالهای مقاله، اثبات ایشان را به‌صورتی کوتاهتر ارائه نمود. به این ترتیب درمدت زمانی کمتر از یک هفته، که در جامعهٔ ریاضی تقریباً بی‌سابقه است، حل مسأله مورد پذیرش جامعهٔ ریاضیدانان قرار گرفت. اگرچه الگوریتم ارائه شده عملاً در مقایسه با الگوریتمهای موجود در زمینهٔ تشخیص اول‌بودن یک عدد طبیعی، تعداد عملیات بیشتری را باید انجام دهد و در نتیجه سرعت اجرای کمتری دارد اما مزیت آن نسبت به این الگوریتمها در دو ویژگی چندجمله‌ای‌بودن طول زمان اجرای عملیات و قطعیتی‌بودن آن است، به‌خصوص که الگوریتمهای قبلی یا دارای ویژگی احتمالاتی بودند و یا بر فرضهای اثبات‌نشده‌ای، مانند حدس تعمیم‌یافتهٔ ریمان متکی بودند. این تمایزها به لحاظ نظری حائز اهمیت هستند وگرنه الگوریتمهای احتمالاتی سریعتری [۱۴] وجود دارند که اول‌بودن یک عدد را با احتمال زیادی و در زمان بسیار کوتاهتری تعیین می‌کنند.

در این مقاله پس از طرح مسأله، تلاشهای انجام‌گرفته برای حل آن را به‌ترتیب تاریخی معرفی و بررسی می‌کنیم و سپس به الگوریتم جدید اگراوال‌ـکایال‌ـساکسنا می‌پردازیم.

صورت مسأله

فرض کنیم $\mathbb{N}$ مجموعهٔ اعداد طبیعی و Primes مجموعهٔ اعداد اول باشد در این صورت

۱. آیا می‌توان روشی کارآمد برای تعیین عضویت یک عضو دلخواه n از $\mathbb{N}$ در مجموعهٔ Primes پیدا کرد؟

1. M. Agrawal 2. N. Saxena 3. N. Kayal 4. C. Pomerance
5. H. W. Lenstra 6. D. Bernstein

به عبارت دیگر آیا می‌توان روش قاعده‌مندی پیدا کرد که به همهٔ پرسشهایی از یک نوع، مثلاً «آیا n عددی اول است؟» پاسخ دهد؟

مسألهٔ دیگری که همواره همراه با این مسأله طرح می‌شود این است که:

۲. آیا می‌توان روشی کارآمد برای تجزیهٔ n به حاصلضربی از عناصر Primes پیدا کرد؟

این دو مسأله از آن جهت اهمیت دارند که ارتباط نزدیکی با بسیاری از مسائل مربوط به اعداد صحیح دارند و علی‌رغم اینکه قدمت آنها به قبل از میلاد مسیح می‌رسد، هنوز یکی از حوزه‌های فعال تحقیقات ریاضی را تشکیل می‌دهند و هر روز خبر موفقیتی تازه در این زمینه می‌رسد، به‌خصوص که امروزه از اعداد اول بزرگ برای به رمز درآوردن اطلاعات و انتقال آن از طریق خطوط مخابراتی از نقطه‌ای به نقطه دیگر استفاده می‌کنند زیرا دشواری تجزیهٔ اعداد صحیح بزرگ، به نوعی ایمنی اطلاعات رمزی شده را در مقابل رمزگشایی آن توسط افراد غیرمجاز افزایش می‌دهد.

اقلیدس در کتاب اصول خود، این دو مسأله را از دیدگاه وجودی بررسی می‌کند و گاوس که به شهریار ریاضیات معروف است، در کتاب تحقیقات حسابی خود [۱۱، ص ۳۹۶] این دو مسأله را چنین توصیف می‌کند:

«مسألهٔ تشخیص اعداد اول از اعداد مرکب و تجزیهٔ اعداد مرکب به عوامل اول، یکی از مهمترین و مفیدترین مسائل حساب شناخته شده است. هندسه‌دانان قدیم و جدید چندان به این مسأله پرداخته‌اند که صحبت مفصل در بارهٔ آن بسیار زائد خواهد بود. معهذا باید اعتراف کنیم که تمامی روشهایی که تاکنون پیشنهاد شده، یا به حالتهای خیلی خاصی محدود بوده‌اند و یا آنقدر پرزحمت و کسل‌کننده‌اند که حتی برای اعدادی که از حد جداول ساخته‌شده توسط محاسبه‌گران حرفه‌ای تجاوز نمی‌کنند، یعنی برای اعدادی که نیازی به روشهای هوشمندانه برای تشخیص اول‌بودنشان نیست، صبر و حوصلهٔ زیادی از محاسبه‌گران حرفه‌ای طلب می‌کنند و آنها را به زحمت می‌اندازند. این روشها را به سختی می‌توان برای اعداد بزرگتر به‌کار برد. به علاوه شأن علم ایجاب می‌کند که به هر وسیله ممکن که بشود، برای مسأله‌ای چنین زیبا و چنین پرآوازه راه‌حلی کشف کرد.»

از طرف دیگر ماهیت الگوریتمی راه‌حل‌های مورد نظر برای دو مسألهٔ فوق آنها را در کانون توجه دانشمندان علوم نظری رایانه نیز قرار داده است، به‌خصوص که این دو مسأله در ارتباط با یک سؤال اساسی این حوزه، یعنی این سؤال قرار دارند که: آیا می‌توان مسائلی طرح کرد که از محدودهٔ توانایی روشهای ماشینی خارج باشد؟ و این دانشمندان طبعاً به دنبال یافتن راه‌حل مناسبی برای این دو مسأله هستند. مسائلی از این نوع، به‌طور مشخص در نظریهٔ محاسبه‌پذیری[1] و پیچیدگی[2] محاسبه مورد بررسی قرار می‌گیرند که هدف آنها پیداکردن شاخصی کمّی برای الگوریتمهایی است که برای حل مسائل ارائه می‌شوند تا بتوان به این وسیله الگوریتمهای مختلفی را که برای یک مسأله ارائه می‌شوند با هم مقایسه کرد. به دلیل محدودیت منابع، به‌خصوص دو منبع زمان لازم برای اجرای الگوریتم و رسیدن به نتیجه و همچنین فضای حافظهٔ لازم برای ذخیره‌سازی نتایج در هر مرحله از اجرا، الگوریتمهایی که از این دو منبع کمتر استفاده کنند، بر آنهایی که لااقل از یکی از این دو منبع بیشتر استفاده می‌کنند، ارجحیت دارند. به این لحاظ، زمان لازم برای اجرای دستورالعملهای یک الگوریتم برروی ورودی آن، یک شاخص مهم الگوریتم به‌شمار می‌رود و معمولاً به صورت تابعی از اندازهٔ ورودی الگوریتم، که دامنه‌اش مجموعهٔ اعداد طبیعی است، بیان می‌شود. در اینجا زمان برحسب ثانیه یا چرخهٔ پردازشگر یک رایانهٔ خاص بیان نمی‌شود، بلکه منظور از آن، تعداد عملیات مقدماتی (بیتی) یا مراحل مورد نیاز برای اجرای یک الگوریتم است. الگوریتمها را برحسب تابع اخیر به دو دستهٔ زمان‌چندجمله‌ای[1] و زمان‌نمایی[2] تقسیم می‌کنند و مسائلی که زمان اجرای نمایی دارند، مسائل دشوار نامیده می‌شوند. علاوه بر سرعت، دو ویژگی زیر نیز در بررسی آنها حائز اهمیت هستند.

- درستی (یا قطعیت). یعنی الگوریتم باید بتواند خروجی صحیح به‌دست دهد.
- عمومیت. یعنی الگوریتم باید بتواند برای همهٔ ورودیها (اعداد) و نه فقط انواع خاصی از آنها اجرا شود و خروجی را تعیین کند.

مسائل مورد بحث در این نظریه براساس ویژگیهای الگوریتمهایی که برای حلشان ارائه شده رده‌بندی می‌شوند. برای معرفی بعضی از آنها که در این مقاله مورد نیازند، فرض کنیم برنامهٔ رایانه‌ای دارای دستورالعمل $GOTO\ L\ OR\ L'$، که L و L' شماره‌های سطر برنامه هستند، باشد. همچنین فرض کنیم با پرتاب یک سکهٔ سالم تصمیم می‌گیریم که کدام‌یک از سطرها باید برای اجرا انتخاب شوند. ردهٔ NP[3] (غیرقطعیتی و زمان چندجمله‌ای) متشکل از تمامی زیرمجموعه‌های A از $\mathbb{N}$ است به‌طوری که برنامه یک ورودی N را می‌پذیرد و دارای ویژگیهای زیر است:

۱. برای یک عدد صحیح و مثبت d، برنامه پس از اجرای n^d مرحله متوقف می‌شود. در اینجا n، طول عدد ورودی N در یک دستگاه شمار، معمولاً دودویی، است.

۲. اگر N عضو A نباشد خروجی الگوریتم «نه» است.

۳. اگر N عضو A باشد، آنگاه به‌ازای حداقل یک دنباله از پرتابها، خروجی «بله» است.

ردهٔ RP[4] (تصادفی و زمان چندجمله‌ای) مشابه ردهٔ NP است، منتهی به جای **۳** باید قرار داد:

۳′. اگر N در A باشد آنگاه به‌ازای حداقل ۵۰٪ دنباله‌های پرتاب سکه، خروجی «بله» است.

رده‌های مسائل P و EXP متشکل از مجموعه‌هایی هستند که عضویت در آنها به کمک برنامه‌هایی که روی هر رایانه اجرا می‌شوند تعیین می‌شود [۱۶]. به عبارت دیگر از دستور $GOTO$ احتمالاتی استفاده نمی‌کنند و زمان اجرای آنها نیز تابعی چندجمله‌ای (در مورد ردهٔ P) و تابعی نمایی (در مورد ردهٔ EXP) از طول عدد ورودی است. بنابر تعریف، رده‌های مسائل

1. computability 2. complexity

1. Polynomial time 2. Exponential time

3. Non-deterministic Polynomial time

4. Random Polynomial time

فوق رابطهٔ زیر را با یکدیگر دارند.

$$\mathrm{P} \subseteq \mathrm{RP} \subseteq \mathrm{NP} \subseteq \mathrm{EXP}$$

روشن است که $\mathrm{P} \neq \mathrm{EXP}$، اما $\mathrm{P} \neq \mathrm{NP}$ یک حدس حل‌نشده است که بنیاد کلی[1] در سال ۲۰۰۰ جایزه‌ای یک میلیون دلاری برای کسی که درستی یا نادرستی آن را ثابت کند تعیین کرده است [۱۷].

با این مقدمه، در قسمت بعد، از هر رده یک الگوریتم برای مسألهٔ اول‌بودن ارائه می‌کنیم و در پایان، الگوریتم اگراوال‌ـکایال‌ـساکسنا (AKS) را شرح می‌دهیم. بجز دو الگوریتم مقدماتی ۱ و ۲، هستهٔ اصلی بقیهٔ الگوریتمها را قضیهٔ کوچک فرما تشکیل می‌دهد. در این الگوریتمها، اگر ورودی اول باشد، خروجی الگوریتم دلالت بر اول‌بودن ورودی می‌کند. اما ممکن است ورودی مرکب باشد ولی خروجی الگوریتم دلالت بر اول‌بودن ورودی بنماید که آشکارا نادرست است. به همین دلیل در بخشی از این الگوریتمها دستورالعملهایی وجود دارد که این خروجی به ظاهر درست ولی واقعاً نادرست به دست نیاید و در واقع قسمت مشکل اثبات درستی این الگوریتمها را تشکیل می‌دهد. این تلاش برای اثبات درستی الگوریتمها نشان می‌دهد که تا چه اندازه نظریه‌های قوی ریاضی می‌توانند راهگشای ارائه راه‌حل‌های ساده برای انجام محاسبات درست شوند.

آزمونهای اول‌بودن

بجز الگوریتم تقسیم و الگوریتم غربال اراتستن، الگوریتمهایی را که برای آزمون اول‌بودن یک عدد طبیعی ابداع شده‌اند می‌توان به سه گروه عمدهٔ زیر تقسیم کرد.

۱. الگوریتمهای احتمالاتی و الگوریتمهایی که با فرض درستی حدس تعمیم‌یافتهٔ ریمان ساخته شده‌اند، یا به‌عبارت دیگر الگوریتمهای شرطی.

۲. الگوریتمهایی که برای آزمون اول‌بودن اعداد خاصی، مانند اعداد فرما $F_n = ۲^{۲^n} + ۱$ یا اعداد مرسن $M_p = ۲^p - ۱$ و اعداد به‌صورت $N = f \cdot ۲^n + ۱$، ساخته شده‌اند [۱۴].

۳. الگوریتمهایی که براساس برخی خواص خمهای بیضوی و یا نظریهٔ اعداد جبری در طی ۲۰ سال گذشته ابداع شده و گسترش یافته‌اند [۷].

چون الگوریتم AKS در اساس مشابهت بسیاری با الگوریتمهای نوع ۱ دارد، در اینجا تنها در بارهٔ برخی از الگوریتمهای نوع ۱، الگوریتم تقسیم و الگوریتم اراتستن توضیح می‌دهیم.

۱. **الگوریتم تقسیم.** ساده‌ترین آزمون اول‌بودن یک عدد طبیعی براساس تعریف عدد اول است یعنی عدد فرد n بر اعداد فرد ۲ تا $[\sqrt{n}]$ تقسیم می‌شود و از روی ماندهٔ تقسیم می‌توان تعیین کرد که آیا n اول است یا خیر. این روش اول‌بودن یک عدد را با قطعیت تعیین می‌کند و برای هر عدد طبیعی قابل اجراست. اما با بزرگترشدن عدد n، چون تعداد عملیات تقسیم افزایش می‌یابد، سرعت رسیدن به نتیجه کاهش می‌یابد. در واقع تعداد عملیات بیتی لازم برای رسیدن به نتیجه تابعی نمایی از طول عدد n در یک دستگاه شماراست. به این ترتیب، با این الگوریتم، مسألهٔ اول‌بودن در ردهٔ EXP قرار می‌گیرد. به‌عنوان مثال اگر n یک عدد s رقمی در پایهٔ ۱۰ باشد آنگاه $n > ۱۰^{s-۱}$ و یا $\sqrt{n} > \sqrt{۱۰^{s-۱}}$ خواهد بود و با ۱۰۰ برابرشدن n، تعداد تقسیمهایی که باید انجام داد نیز لااقل ۱۰ برابر بیشتر می‌شود. از طرف دیگر، مزیت این روش در این است که یک تجزیهٔ نابدیهی و درست از عدد n را به‌دست می‌دهد و به این ترتیب الگوریتم فوق می‌تواند مسألهٔ ۲ را نیز پاسخ دهد، اما نه پاسخی سریع.

۲. **الگوریتم غربال اراتستن.** این الگوریتم نیز روشی برای تعیین اول‌بودن یک عدد طبیعی به‌دست می‌دهد، منتهی چون در عمل تقسیم، تمام مضارب نابدیهی یک عدد اول بین ۲ تا $[\sqrt{n}]$ را حذف می‌کند تعداد تقسیمها بسیار کاهش می‌یابد و تعداد عملیات بیتی لازم برای رسیدن به نتیجه دیگر نمایی نخواهد بود. این الگوریتم اول‌بودن یک عدد طبیعی را با قطعیت تعیین می‌کند و برای همه اعداد طبیعی قابل اجراست. مراحل این الگوریتم را می‌توان به‌صورت زیر مرتب کرد.

ورودی: یک عدد طبیعی n
خروجی: اول‌بودن یا مرکب‌بودن n
۱. L را برابر فهرست تمامی اعداد صحیح بین ۲ تا $[\sqrt{n}]$ قرار بده. همچنین قرار بده $F = \emptyset$ و $\mathrm{P} = \emptyset$.
۲. m، کوچکترین عدد فهرست L را انتخاب کن. اگر m یک مقسوم‌علیه n بود، قرار بده $F = F \cup \{m\}$ و $\mathrm{P} = \mathrm{P}$. در غیر این صورت $F = F$ و $\mathrm{P} = \mathrm{P} \cup \{m\}$.
۳. تمامی مضارب عدد m را در فهرست L حذف کن و فهرست حاصل را برابر L' قرار بده.
۴. اگر $L' \neq \emptyset$، قرار بده $L = L'$ و دوباره از مرحلهٔ ۲ شروع کن.
۵. اگر $L = \emptyset$ و اگر $F = \emptyset$، آنگاه n عددی اول است و اگر $F \neq \emptyset$، آنگاه F مجموعهٔ تمام مقسوم‌علیه‌های اول n را به‌دست می‌دهد. P نیز مجموعهٔ تمام اعداد نسبت به n اول کمتر از $[\sqrt{n}]$ را به‌دست می‌دهد.

اگرچه زمانی که برای هر عدد m صرف می‌شود تقریباً برابر $O(\log \log m)$ است و این الگوریتم در مجموع نسبت به الگوریتم تقسیم خیلی سریعتر به خروجی الگوریتم نزدیک می‌شود، اما چون میزان فضای حافظهٔ لازم برای ذخیرهٔ اعضای P و F، هنگامی که n بزرگ و بزرگتر می‌شود به‌صورت نمایی رشد می‌کند و از محدودهٔ قابل قبول فضای حافظهٔ رایانهٔ نظری که برای اجرای الگوریتم در نظر گرفته می‌شود تجاوز خواهد کرد، عملاً قابل استفاده نیست.

۳. **الگوریتم فرما.** سومین الگوریتم آزمون اول‌بودن برپایهٔ قضیهٔ کوچک‌فرما ساخته شده است.

قضیهٔ ۱ (قضیهٔ کوچک‌فرما). اگر n یک عدد اول باشد و a یک عدد صحیح به طوری که $(a, n) = ۱$، آنگاه (پیمانهٔ n) $a^{n-۱} \equiv ۱$.

ماندهٔ تقسیم $a^{n-۱}$ بر n را می‌توان با استفاده از روابط زیر به شیوه‌ای

1. Clay

مؤثر به‌دست آورد. فرض کنیم n یک عدد طبیعی باشد و $n-1$ را در پایهٔ ۲ بتوان به‌صورت

$$n-1=b_0+b_1 2+b_2 2^2+b_3 2^3+\cdots+b_k 2^k, b_i\in\{0,1\}, 0\le i\le k$$

نوشت. در این صورت

$$a^{n-1}=a^{b_0}\times a^{b_1 2}\times a^{b_2 2^2}\times\cdots\times a^{b_k 2^k}$$

این رابطه نشان می‌دهد که برای محاسبهٔ ماندهٔ تقسیم a^{n-1} بر n کافی است با مربع‌کردن متوالی رابطهٔ (پیمانهٔ n) $a^{2^i}\equiv r$ و پیداکردن ماندهٔ جدید، ماندهٔ تقسیم a^{n-1} بر n را به‌دست آورد. این عملیات در زمانی چندجمله‌ای، معادل $O(\log^3 n)$ انجام خواهد گرفت.

الگوریتمی را که برپایهٔ مشاهدات فوق ساخته می‌شود می‌توان به شکل زیر صورت‌بندی کرد:

ورودی: یک عدد طبیعی n

خروجی: n مرکب است یا معلوم نیست که اول است.

۱. یک عضو a از $\{2,\ldots,n-1\}$ را انتخاب کن.

۲. (a,n) را محاسبه کن. اگر $(a,n)>1$ آنگاه n مرکب است.

۳. اگر $(a,n)=1$ آنگاه ماندهٔ تقسیم a^{n-1} به پیمانهٔ n را حساب کن. اگر این مانده بزرگتر از یک بود آنگاه n مرکب است. در غیر این‌صورت، نمی‌توان با قاطعیت گفت که n اول است.

اشکال به کارگیری این الگوریتم در این است که چون عکس قضیهٔ فوق درست نیست، الگوریتم نمی‌تواند به‌عنوان آزمونی برای اول‌بودن مورد استفاده قرار گیرد. به‌عنوان مثال، اگرچه $(7,25)=1$ و (پیمانهٔ ۲۵) $7^{24}\equiv 1$ اما ۲۵ عددی اول نیست. به این ترتیب این الگوریتم علی‌رغم اینکه زمان‌چندجمله‌ای و برای همهٔ اعداد طبیعی قابل اجراست، ویژگی قطعیتی ندارد.

هر عدد طبیعی مرکبی nای که رفتاری شبیه رفتار اعداد اول در قضیهٔ کوچک‌فرما داشته باشد یعنی به‌ازای عدد صحیح a که $(a,n)=1$، در حکم قضیه صدق کند، به عدد شبه‌اول[1] در پایه a موسوم است. برای یک عدد ثابت a، مشاهده‌شده که اعداد شبه‌اول در پایهٔ a خیلی نادرند. حتی در مقایسه با اعداد اول، تعدادشان در بازه‌های محدود خیلی کمتر از تعداد اعداد اول در همان بازه است. به‌عنوان مثال تعداد اعداد اول کمتر از 10^{10} برابر ۴۵۵۰۵۲۵۱۲ است در حالی‌که در این فاصله ۱۴۸۸۴ عدد شبه‌اول در پایهٔ ۲ وجود دارد [۱۳]. یعنی به‌طور نسبی، به‌ازای هر ۱۰۰۰ عدد اول کوچکتر از 10^{10}، ۳ عدد مرکب وجود دارند که به‌ازای $a=2$ در قضیهٔ کوچک‌فرما صدق می‌کنند. حتی اعداد مرکب nای وجود دارند که در تمام پایه‌های a، که $(a,n)=1$، در قضیهٔ کوچک فرما صدق می‌کنند. به‌عنوان مثال، عدد $561=3\times 11\times 17$ به‌ازای هر عدد صحیح a که $(a,n)=1$، در قضیهٔ کوچک‌فرما صدق می‌کند. چنین اعدادی را اعداد کارمایکل[2] می‌نامند. ثابت شده است که تعداد این اعداد بی‌پایان است [۱].

۴. **الگوریتم میلر**[3]. این الگوریتم در سال ۱۹۷۶ توسط میلر ارائه شد [۱۲]. سؤال زیر می‌تواند توجیه‌کنندهٔ ساختار این الگوریتم باشد.

1. pseudoprime 2. Carmichael numbers 3. G. L. Miller

سؤال: مرکب‌بودن یا اول‌بودن n چه نقشی در جوابهای معادلهٔ همنهشتی (پیمانهٔ n) $a^2\equiv 1$ می‌تواند داشته باشد؟

فرض کنیم n یک عدد اول باشد. آنگاه معادلهٔ (پیمانهٔ n) $a^2\equiv 1$ تنها دو جواب ± 1 دارد. از طرف دیگر اگر n حاصلضرب k عامل اول فرد باشد، آنگاه بنابر قضیهٔ ماندهٔ چینی معادله اخیر دارای 2^k جواب خواهد بود. حال اگر (پیمانهٔ n) $a^{n-1}\equiv 1$، آنگاه واضح است که $a^{\frac{n-1}{2}}$ یک ریشهٔ دوم به پیمانهٔ n خواهد بود. بنابراین اگر (پیمانهٔ n) $a^{\frac{n-1}{2}}\not\equiv\pm 1$، آنگاه n یک عدد مرکب خواهد بود. به‌عنوان مثال عدد مرکب ۵۶۱ که در آزمون فرما صدق می‌کند در پایهٔ ۷ در رابطهٔ (پیمانهٔ ۵۶۱) $7^{560}\equiv 1$ صادق است ولی (پیمانهٔ ۵۶۱) $7^{280}\not\equiv 1$؛ به این ترتیب ۵۶۱ عددی مرکب است. این استدلال نشان می‌دهد که می‌توان نقص الگوریتم ۳ را در مورد ورودیهایی که اول نیستند ولی در قضیهٔ کوچک‌فرما صدق می‌کنند، تا اندازه‌ای مرتفع کرد. برای این کار فرض کنیم n عددی فرد باشد و $n-1=2^s m$ که $(2,m)=1$. این تجزیه در زمانی چندجمله‌ای صورت می‌گیرد. اگر $(a,n)=1$ و (پیمانهٔ n) $a^m\equiv 1$ یا به‌ازای یک i، به‌طوری که $1\le i\le s-1$، (پیمانهٔ n) $a^{2^i m}\equiv -1$، آنگاه n اول است، یا نمی‌توان از این نتیجه به مرکب‌بودن آن پی برد. و اگر به‌ازای a، با $(a,n)=1$ یکی از تساویهای فوق برقرار نشود، آنگاه n عددی مرکب است. براساس مشاهدهٔ فوق الگوریتم زیر را می‌توان ساخت. اما قبل از بیان دستورالعملهای الگوریتم و برای سادگی، به‌ازای $a\in\{2,3,\ldots,n-1\}$، دنبالهٔ

$$a^m(\text{پیمانهٔ } n), a^{2m}(\text{پیمانهٔ } n), a^{2^2 m}(\text{پیمانهٔ } n),\ldots,$$
$$a^{2^{s-1}m}(\text{پیمانهٔ } n)\quad (*)$$

را یک دنبالهٔ نوع اول می‌نامیم هرگاه تمامی عناصر آن یک باشد و یا یکی از عناصر ماقبل آخر آن برابر -1 باشد. در غیر این صورت آن را یک دنبالهٔ نوع دوم می‌نامیم. الگوریتم به‌صورت زیر است.

ورودی: یک عدد طبیعی فرد

خروجی: عدد n مرکب است یا معلوم نیست که اول است.

۱. قرار بده $S=\{2,3,\ldots,n-1\}$.

۲. قرار بده $n-1=2^s m$ که $(m,2)=1$.

۳. یک عضو a از S انتخاب کن.

۴. اگر $(a,n)\neq 1$ آنگاه n مرکب است.

۵. اگر $(a,n)=1$ با a، دنبالهٔ $(*)$ را تشکیل بده.

۶. اگر این دنباله از نوع دوم بود، آنگاه n یک عدد مرکب است.

۷. اگر این دنباله از نوع اول بود، آنگاه معلوم نیست که n اول است.

عملیات بیتی مرحلهٔ ۲، چندجمله‌ای است. همچنین تعداد عملیات بیتی لازم برای هر a برابر $O(\log^3 n)$ است. بنابراین الگوریتم فوق یک الگوریتم زمان چندجمله‌ای است. چون این الگوریتم نیز براساس قضیهٔ کوچک‌فرما ساخته شده، همان نقص الگوریتم فرما را دارد. اگر ورودی مرکب و فرد n، به‌ازای یک aی عضو S با شرط $(a,n)=1$، دنباله‌ای از نوع اول تولید

کند، یک عدد شبه‌اول قوی[1] در پایهٔ a نامیده می‌شود. به‌عنوان مثال عدد $۲۰۴۷ = ۲۳ \times ۸۹$ در پایهٔ ۲، یک دنباله از نوع اول تولید می‌کند و لذا یک عدد شبه‌اول قوی در پایهٔ ۲ است در [۱۳] ثابت می‌شود تعداد اعداد شبه‌اول قوی در یک پایهٔ معین a نامتناهی است. از طرف دیگر، نشان داده می‌شود [۶. ص. ۸۱] که هیچ عدد مرکبی مانند n وجود ندارد که به‌طور همزمان در همهٔ پایه‌های a، با $(a,n) = ۱$، شبه‌اول قوی باشد. به‌عنوان مثال در [۱۳] نشان داده می‌شود که تنها یک عدد $n < ۲۵ \times ۱۰^۹$ وجود دارد که به‌طور همزمان در پایه‌های ۲، ۳، ۵ و ۷ شبه‌اول قوی است اما در پایهٔ ۱۱، دنباله‌ای از نوع دوم تولید می‌کند. به این ترتیب در استفاده از الگوریتم میلر تنها تعداد کمی از اعدادی که نسبت به ورودیِ اول‌اند باید برای تشکیل دنبالهٔ مانده‌ها و تصمیم‌گیری در بارهٔ اول‌بودن n مورد استفاده قرار گیرند و بنابراین تعداد عملیات لازم برای تشخیص اول‌بودن یا مرکب‌بودن n بسیار کاهش می‌یابد.

اما چگونه می‌توان این اعداد را در مجموعهٔ $\{۲,۳,\dots,n-۱\}$ شناخت و انتخاب کرد؟

فرض کنیم n یک عدد اول فرد باشد و a یک عضو $\{۲,۳,\dots,n-۱\}$. همچنین، فرض کنیم $n-۱ = ۲^s m$ که $(m,۲) = ۱$ و $a^{n-۱} \equiv ۱ (n$ (پیمانهٔ $n)$ یا $n | a^{n-۱} - ۱$. در نتیجه n باید یکی از عوامل سمت راست تساوی

$$a^{n-۱} - ۱ = (a^m - ۱)(a^m + ۱)(a^{۲m} + ۱)(a^{۴m} + ۱)\dots (a^{۲^{s-۱}m} + ۱)$$

را بشمارد. بنابراین دنبالهٔ (*) دنباله‌ای از نوع اول خواهد بود.

حال فرض کنیم $n = p^j$ که $j > ۱$ و p یک عدد اول فرد است. می‌دانیم یک عضو a از $\{۲,۳,\dots,n-۱\}$ وجود دارد به‌طوری که $(a,n) = ۱$ و $a^{\varphi(n)} \equiv ۱ (n$ (پیمانهٔ $n)$، ولی به‌ازای هر $d < \varphi(n)$، $a^d \not\equiv ۱ (n$ (پیمانهٔ $n)$. چنین عنصری را یک ریشهٔ اولیه به پیمانهٔ n می‌نامند. با این فرض، به‌ازای هر ریشهٔ اولیهٔ a، $a^{n-۱}$ همنهشت با ۱ به پیمانهٔ n نخواهد بود. زیرا اگر $a^{n-۱}$ همنهشت با یک به پیمانهٔ n باشد آنگاه $p^{j-۱} | p^j - ۱$ $\varphi(n) = (p-۱)p^{j-۱}$ که یک تناقض است. بنابراین در این حالت با انتخاب یک ریشهٔ اولیه، دنبالهٔ (*) از نوع دوم خواهد بود و الگوریتم به درستی مرکب‌بودن n را نتیجه می‌دهد. حال فرض کنیم n یک عدد مرکب فرد باشد که توانی از یک عدد اول فرد نیست. هرگاه هر یک از عوامل ضرب n، لااقل یکی از عوامل سمت راست تساوی فوق را بشمارد، دنبالهٔ (*) از نوع اول خواهد بود و n نیز در رابطهٔ (پیمانهٔ $n)$ $a^{n-۱} \equiv ۱$ صدق خواهد کرد. در این حالت نمی‌توان از روی دنبالهٔ (*) به اول‌بودن یا مرکب‌بودن n پی برد. با این فرض، این وضعیت نمی‌تواند برای تمامی عناصر مجموعهٔ $\{۲,۳,\dots,n-۱\}$ پیش آید زیرا

قضیهٔ ۲ [مرجع ۶. ص. ۸۲]. فرض کنیم n یک عدد مرکب فرد با لااقل دو عامل اول p و q باشد که $p = ۲^{s_۱} m_۱ + ۱$ و $q = ۲^{s_۲} m_۲ + ۱$ و $(m_۱,۲) = (m_۲,۲) = ۱$. گیریم $s_۱, s_۲ \geq ۱$ و همچنین $s_۱ \leq s_۲$.

نیز فرض کنیم a عددی صحیح با شرط $(a,n) = ۱$ باشد، گیریم ماندهٔ a به پیمانهٔ p مربع کامل باشد و به پیمانهٔ q مربع کامل نباشد، در این صورت دنبالهٔ (*) دنباله‌ای از نوع دوم خواهد بود.

برای آزمودن دو شرط آخر قضیهٔ اخیر از محک زیر (محک اویلر) می‌توان استفاده کرد.

قضیهٔ ۳ [مرجع ۶. ص. ۷۹]. فرض کنیم p یک عدد اول باشد، همچنین فرض کنیم a یک عدد صحیح باشد که بر p بخش‌پذیر نیست، آنگاه (پیمانهٔ p) $a^{\frac{p-۱}{۲}} \equiv ۱$ اگر و تنها اگر یک عدد صحیح t وجود داشته باشد به طوری‌که (پیمانهٔ p) $a \equiv t^۲$.

در قضیهٔ ۳، a را می‌توان به کمک قضیهٔ ماندهٔ چینی به‌دست آورد. اما این قضیه مبتنی بر فرضِ دانستنِ لااقل دو عامل اول n است که چنین aای را به‌دست می‌دهند و اگر قبلاً این دو عامل را می‌شناختیم، خود به خود مسألهٔ مرکب‌بودن n پاسخی مثبت داشت. به ناچار باید از ابزار دیگری برای پیداکردن a استفاده کرد.

تابع $\chi : \mathbb{Z}_n \to \{-۱,۱\}$ را، که $\mathbb{Z}_n$ گروه جمعی مانده‌های تقسیم به پیمانهٔ n است، به‌صورت زیر تعریف می‌کنیم

$$\chi(a) = \begin{cases} (a/q) & s_۱ < s_۲ \\ (a/q)(a/p) & s_۱ = s_۲ \end{cases}$$

که $(a/q) = ۱$ هرگاه ماندهٔ a به پیمانهٔ q مربع کامل باشد و $(a/q) = -۱$ هرگاه این مانده مربع کامل نباشد. روشن است که χ یک همریختی گروههاست و بنابر قضیهٔ ۲، aای وجود دارد که $\chi(a) = -۱$. بنابراین χ یک همریختی نابدیهی است. در نتیجه، بنابر قضیهٔ ۲، برای نیمی از اعداد بین ۱ تا n، عدد n یک عدد شبه‌اول قوی در پایهٔ آنها نخواهد بود. یا به عبارت دیگر با انتخاب چنین اعدادی، آزمون میلر حکم به مرکب‌بودن n خواهد داد. میلر [۱۲] با مشاهدهٔ اینکه گروه ضربی $\{۱,-۱\}$ یک زیرگروه ضربی $\mathbb{C}^*$ است و همریختی فوق یک همریختی از $\mathbb{Z}_n$ در $\mathbb{C}^*$ تعریف می‌کند، به کمک قضیه‌ای از انکنی [۴] ثابت کرد که در صورت پذیرش درستی حدس تعمیم‌یافتهٔ ریمان[1] (ERH)، یک ثابت مثبت c وجود دارد به طوری‌که به‌ازای هر مشخصهٔ ناثابتی $(\neq ۱)$ به پیمانهٔ n یک عدد صحیح و مثبت a موجود است که $a < c \log^۲ n$ و $(a,n) = ۱$ و $\chi(a) = -۱$. به این ترتیب با کمک قضیهٔ انکنی نیازی به انتخاب تمامی اعضای $\{۲,۳,\dots,n-۱\}$ برای تشکیل دنبالهٔ مانده‌های (*) به‌منظور تشخیص مرکب‌بودن و یا اول‌بودن n نیست و کافی است این دنباله را برای اعضای $\{۲,۳,\dots,[c\log^۲ n]\}$

1. strong pseudoprime

۱. حدس تعمیم‌یافتهٔ ریمان (ERH). ریمان حدس زد که صفرهای تابع زتا که از توسیع $\zeta(s) = \sum_{n=۱}^{\infty} \frac{۱}{n^s}$ به $\mathrm{Re}(s) > ۰$ به‌دست می‌آید روی خط $\mathrm{Re}(s) = \frac{۱}{۲}$ قرار دارند. درستی یا نادرستی این حدس نیز ازجمله مسائل مشکل نظریهٔ اعداد و یکی از ۷ مسألهٔ یک میلیون دلاری بنیاد کلی در سال ۲۰۰۰ است [۱۶].

اگر به جای سری فوق، سری $L(\zeta,\chi) = \sum_{\substack{a=۱ \\ (a,n)=۱}}^{\infty} \frac{\chi(a)}{a^\zeta}$ را که در آن χ یک تابع مشخصه از $\mathbb{Z}_n$ در $\mathbb{C}^*$ است، در نظر بگیریم، این حکم را که صفرهای سری فوق روی خط $\mathrm{Re}(\zeta) = \frac{۱}{۲}$ قرار می‌گیرند حدس تعمیم‌یافتهٔ ریمان می‌نامند [۱۰].

تشکیل داد. چون تعداد عملیات بیتی الگوریتم برای aی معین $O(\log^{۳} n)$ است و $a < c\log^{۲} n$، پس تعداد عملیات بیتی الگوریتم فوق حداکثر $\log^{۵} n$ خواهد بود. به این ترتیب الگوریتم میلر، با فرض درستی حدس تعمیم‌یافتهٔ ریمان، الگوریتمی زمان‌چندجمله‌ای و قطعیتی خواهد بود. به عبارت دیگر با این الگوریتم، مسألهٔ Primes یک مسألهٔ P خواهد بود. ولی نقص این الگوریتم مشروط‌بودن آن به درستی حدس تعمیم‌یافتهٔ ریمان است.

در [۸] رهیافتی دیگر به مسألهٔ فوق، منتهی بدون استفاده از ERH ارائه شده است.

۵. الگوریتم احتمالاتی رابین. همان‌طورکه قبلاً گفته شد، الگوریتم احتمالاتی الگوریتمی است که ممکن است زمان اجرای خیلی طولانی یا نامحدود داشته باشد. البته با شرطی‌کردن تعداد تکرارهای الگوریتم، اجرای الگوریتم پس از برآورده‌شدن شرط متوقف می‌شود و نتیجهٔ به‌دست آمده با احتمال زیادی درست خواهد بود. اما اگر قبل از برآورده‌شدن شرط، الگوریتم به جواب برسد، نتیجهٔ به‌دست آمده صحیح خواهد بود.

رابین مشاهده کرد که اگر n عددی اول یا توانی از یک عدد اول نباشد آنگاه معادلهٔ $a^{۲} \equiv ۱ (n)$ ، $(n,a) = ۱$ حداقل دارای چهار ریشه است و ثابت کرد که در الگوریتم میلر هیچ عدد مرکبی نمی‌تواند یک عدد شبه‌اول قوی برای بیش از $\frac{n}{۴}$ انتخابهای a از مجموعهٔ $\{۱, ۲, ۳, \ldots, n-۱\}$ باشد. بنابراین اگر a به تصادف از $\{۱, ۲, ۳, \ldots, n-۱\}$ انتخاب شود احتمال اینکه دنبالهٔ مانده‌ها به پیمانهٔ n، $(*)$، با این انتخابْ دنباله‌ای از نوع اول باشد حداکثر $\frac{۱}{۴}$ و احتمال اینکه از نوع دوم باشد برابر با $\frac{۱}{۴} - ۱$ خواهد بود. رابین براساس استدلال فوق الگوریتم خود را در سال ۱۹۸۰ به‌صورت زیر ارائه داد.

ورودی: عدد فرد n و عدد صحیح و مثبت k.
خروجی: n مرکب است یا با احتمال معینی اول است.
۱. یک عدد به‌تصادف از مجموعهٔ $\{۱, ۲, ۳, \ldots, n-۱\}$ انتخاب کن.
۲. $n-۱$ را تجزیه کن و قرار بده $n-۱ = ۲^{s}m$ به طوری‌که $(m,s) = ۱$.
۳. $b \equiv a^{m} (n)$ (پیمانهٔ n) را محاسبه کن.
۴. اگر $b \equiv \pm ۱ (n)$ (پیمانهٔ n) آنگاه احتمالاً n اول است و اجرا را متوقف کن.
۵. در غیر این‌صورت دنبالهٔ $(*)$ را تشکیل بده. اگر دنباله از نوع دوم بود، n مرکب است و اجرا را متوقف کن.
۶. در غیر این‌صورت به مرحلهٔ ۲ برو و با انتخاب aی دیگر عملیات فوق را انجام بده.
۷. اگر پس از k بار اجرای الگوریتم فوق، در هر بار اجرا، دنبالهٔ $(*)$ از نوع اول بود اجرا را متوقف کن و n احتمالاً اول است.

این الگوریتم علی‌رغم اینکه دارای زمان اجرای چندجمله‌ای است، چون براساس قضیهٔ کوچک فرما ساخته شده همان نقص الگوریتمهای فرما و میلر را دارد و نمی‌تواند قطعیتی باشد. اما چون عدد مرکب n نمی‌تواند در تمام پایه‌های a از دنبالهٔ $(*)$، دنباله‌ای از نوع اول تولید کند احتمال اینکه پس از rبار اجرا متوقف شود حداقل برابر $۴^{-r} - ۱$ است. بنابراین هر چه تعداد تکرارها بیشتر شود دقت نتیجهٔ به‌دست آمده بیشتر خواهد بود. به‌عبارت دیگر می‌توان به دلخواه، دقت نتیجهٔ به‌دست آمده را تعیین کرد. مزیت این الگوریتم بر الگوریتم میلر در استفاده‌نکردن از حدسی اثبات نشده است.

الگوریتم اول‌بودن اگراوال، کایال و ساکسنا (AKS)

الگوریتمهای ۳ و ۴ و ۵، همگی براساس قضیهٔ کوچک فرما ساخته شده‌اند و در آنها به کمک شرایطی اضافی، سعی شده از حالتهایی که ممکن است عدد مرکبی مانند n در قضیه صدق کند پرهیز شود تا به این وسیله بتوانند قطعیتی‌بودن آن را ثابت کنند.

اما رهیافت اگراوال و همکارانش برای رفع این نقیصه، متفاوت با رهیافتهای فوق است و به‌جای محاسبه در $\mathbb{Z}_n$ محاسبات را در حلقهٔ بزرگتر $\mathbb{Z}_n[x]$، یعنی چندجمله‌ای‌های با ضرایب در $\mathbb{Z}_n$، انجام می‌دهند و از محک زیر برای تشخیص اول‌بودن استفاده می‌کنند.

قضیهٔ ۴ [مرجع **۳**]. فرض کنیم n و a دو عدد صحیح مثبت باشند و $(a,n) = ۱$، آنگاه n اول است اگر و تنها اگر

$$(x-a)^{n} \equiv (x^{n}-a)(n \text{ پیمانهٔ})$$

اما عملیات بیتی لازم برای محاسبهٔ ضرایب $(x-a)^{n}$ به‌صورت نمایی رشد می‌کند. اگراوال ابتدا با استفاده از این قضیه یک الگوریتم تصادفی برای آزمون اول‌بودن وضع کرد [**۲**]. اما بعداً با انجام محاسبات در حلقهٔ $\mathbb{Z}_n[x]/(x^{r}-۱)$ که r یک عدد اول کوچک است و به نحو مناسبی انتخاب می‌شود، مشاهده کرد که تعداد این عملیات تا حد یک تابع چندجمله‌ای از ورودی n کاهش می‌یابد. از طرف دیگر با انتخاب $r = ۱$، حلقهٔ $\frac{\mathbb{Z}_n[x]}{(x^{r}-۱)}$ با $\mathbb{Z}_n$ همریخت می‌شود و همان قضیهٔ فرما را نتیجه می‌دهد. اگراوال با انتخاب یک عامل تحویل‌ناپذیر $x^{r}-۱$ مانند $h(x)$ و یافتن یک زیرگروه ضربی میدان $\frac{\mathbb{Z}_n[x]}{(h(x))}$ که p یک شمارندهٔ اول n است و مرتبهٔ آن به‌اندازهٔ کافی بزرگ است، اطلاعاتی راجع به n دست می‌آورد، که در دو قضیهٔ زیر می‌توان آنها را خلاصه کرد.

قضیهٔ ۵ [مرجع **۵**]. فرض کنیم n یک عدد صحیح مثبت باشد و q و r دو عدد اول باشند، همچنین فرض کنیم S یک مجموعهٔ s عضوی از اعداد صحیح باشد، گیریم $q|r-۱$ و $n^{r-۱/q} \not\equiv ۱ (r$ پیمانهٔ$) \not\in \{۰, ۱\}$ و برای هر a و a' متمایز از S رابطهٔ $(n, a-a') = ۱$ برقرار باشد، فرض کنیم $s \in \binom{q+s-۱}{s} \geq n^{۲\lfloor\sqrt{r}\rfloor}$ و برای هر a رابطهٔ $(x+a)^{n} \equiv x^{n}+a (\frac{\mathbb{Z}_n[x]}{(x^{r}-۱)}$ پیمانهٔ$)$ برقرار باشد، آنگاه n توانی از یک عدد اول است.

قضیهٔ ۶ [مرجع **۵**]. فرض کنیم n و r دو عدد صحیح مثبت باشند و S یک مجموعهٔ s عضوی از اعداد صحیح باشد، همچنین فرض کنیم n یک ریشهٔ اولیه به پیمانهٔ r باشد و $(n, a-a') = ۱$ برای همه a و a'های

متمایز عضو S برقرار باشد، گیریم $n^{\lfloor\sqrt{r}\rfloor} \geq \binom{\varphi(r)+s-1}{s}$ و رابطهٔ $(x+a)^n = x^n + a$ در حلقهٔ $\frac{\mathbb{Z}_n[x]}{(x^r-1)}$ برای تمامی aهای عضو S برقرار باشد، آنگاه n توانی از یك عدد اول است.

در اینجا $\lfloor\sqrt{r}\rfloor$ بزرگترین عدد صحیح کوچکتر از $\sqrt{r}$ را نشان می‌دهد. اگراوال و همکارانش پس از اثبات قضایای فوق الگوریتم زیر را ساخته‌اند.

ورودی: عدد صحیح $n > 1$.

خروجی: n مرکب است یا n اول است.

۱. اگر n به‌صورت توانی از یک عدد اول بود خروجی را مرکب قرار بده.

۲. قرار بده $r = 2$.

۳. مادامی که $n > 2$، عملیات را انجام بده.

۴. اگر $1 \neq (n, r)$، خروجی را مرکب قرار بده.

۵. اگر r اول بود بزرگترین عامل اول $r-1$ را بیاب و آن را q بنام { اگر

$$q \geq 4\sqrt{r}\log n \text{، } n^{\frac{r-1}{q}} \not\equiv 1 (r \text{ پیمانهٔ})$$

عملیات را متوقف کن }

۶. وگرنه یکی به r اضافه کن و مجدداً از ۳ شروع کن.

۷. برای a از ۱ تا $2\sqrt{r}\log n$.

اگر $(x-a)^n \not\equiv (x^n-a)((x^r-1, n)$ (پیمانهٔ)، خروجی را مرکب قرار بده.

در صورت انجام کامل عملیات فوق، خروجی یک عدد اول است.

با این الگوریتم، مهمترین قسمت مقالهٔ [۳] اثبات در قضیهٔ زیر است

قضیهٔ ۷ [مرجع **۳**. قضیهٔ ۴.۱]. خروجی الگوریتم فوق اول است اگر و تنها اگر p اول باشد،

که نشان می‌دهد که الگوریتم فوق یک الگوریتم قطعیتی است و

قضیهٔ ۸ [مرجع **۳** قضیه ۵.۱]. پیچیدگی الگوریتم فوق به صورت $O(\text{poly}(\log^{12} n))$ است،

که منظور از $\text{poly}(\log^{12} n)$ یک چندجمله‌ای از متغیر $\log^{12} n$ است. و این قضیه نشان می‌دهد تعداد عملیات بیتی این الگوریتم یک چندجمله‌ای است و به این ترتیب با الگوریتم فوق، مسألهٔ اول‌بودن در ردهٔ P قرار می‌گیرد.

تشکر و قدردانی. نویسندهٔ مقاله لازم می‌داند از آقای دکتر غلامرضا خسروشاهی که متن مقالهٔ اگراوال و همکارانش را در اختیار نویسنده قرار دادند، از آقای دکتر امیر اکبری که راهنماییهای ارزنده‌ای در مورد حدس تعمیم‌یافتهٔ ریمان ارائه کردند، و همچنین از آقای دکتر سیاوش شهشهانی که نقصهای پیش‌نویس مقاله را یادآوری نمودند، سپاسگزاری کند.

مراجع

1. W. R. Alford, A. Granville, C. Pomerance, "There are infinitely many Carmichael numbers", *Ann. Math.*, **140** (1994) 703-722.
2. M. Agrawal, S. Biswas, "Primality and Identity testing via Chinese Remindering", In *Proceeding of Annual IEEE Symposium on Foundation of Computer Science*, (1999) 202-209.
3. M. Agrawal, N. Kayal, N. Saxena, "Primes is in P", preprint (2002), `www.cse.iitk.ac.in/news/primality.html`.
4. N. C. Ankeny, "The least quadratic non-residue", *Ann. Math.*, **55** (1952) 65-72.
5. D. J. Bernstein, "An exposition of the Agrawal-Kayal-Saxena primality-proving theorem", preprint (2002), `cr.yp.tc/papers/aks.pdf`.
6. D. M. Bressoud, *Factorization and Primality*, Springer-Verlag, New York (1989).
7. A. H. Cohen, *A Course in Computational Algebraic Number Theory*, Springer-Verlag, Berlin, Heidelberg (1996).
8. H. Cohen, H. W. Lenstra, "Primality testing, and Jacobi sum", *Math. Comp.*, **42** (1984), 297-330.
9. S. Cook, "The P versus NP problem", preprint (2000), www.claymath.org.
10. J. Dixon, "Factorization and primality tests", *Amer. Math. Monthly*, **91** (1984) 333-352.
11. C. F. Gauss, *Disquisitiones Arithmeticae* (Translated By A. A. Clarke), 2nd Printing, Springer-Verlag (1998).
12. G. Miller, "Reimann's hypothesis and test's for primality", *J. Comp. System Sci.*, **13** (1976) 300-317.
13. C. Pomerance, J. L. Selfridge, S. S. Wagstaff, Jr., "The pseudoprimes to $25 * 10^9$", *Math. Comp.* **35** (1980) 1003-1026.
14. Primepages.org.
15. M. O. Rabin. "Probabilistic algorithms for primality testing", *J. Number Theory*, **12** (1980) 128-138.
16. S. Wagon, "Primality testing", *Math. Intelligencer*, (3) **8** (1986).
17. www.claymath.org.

* * * * * *

* حسن حقیقی، دانشگاه صنعتی خواجه نصیرالدین طوسی

haghighi@kntu.ac.ir