دانشکده مهندسی کامپیوتر

تمرین زنون (تمرین نظریه اعداد، فصل سوم)

امنیت سیستم‌های کامپیوتری

مدرس: دکتر ابوالفضل دیانت

محمدحسین عباسپور، فرزان رحمانی

شماره دانشجویی: ۹۹۵۲۱۴۳۳، ۹۹۵۲۱۲۷۱

نیم سال دوم
سال تحصیلی ۱۴۰۳-۱۴۰۲

۱ پیچیدگی زمانی سریع ترین الگوریتم برای تجزیه اعداد اول چگونه است؟

در واقع، هیچ الگوریتم "سریع" شناخته شده ای برای فاکتورگیری اعداد صحیح بزرگ در یک کامپیوتر کلاسیک وجود ندارد. به همین دلیل است که فاکتورسازی اعداد صحیح یک مشکل محاسباتی سخت در نظر گرفته می‌شود و برای امنیت بسیاری از سیستم‌های رمزنگاری مانند RSA بسیار مهم است. دشواری فاکتورگیری اعداد بزرگ از این واقعیت ناشی می شود که شناخته شده ترین الگوریتم ها در تعداد ارقام عدد فاکتورگیری شده زمان نمایی می گیرند. پیچیدگی زمانی سریعترین الگوریتم شناخته شده برای فاکتورسازی اعداد صحیح به اعداد اول به اندازه عدد ورودی بستگی دارد. الگوریتم های مختلفی وجود دارند که برای اندازه های ورودی مختلف عملکرد بهتری دارند. برای اعداد بسیار بزرگ (اعداد با صدها یا هزاران رقم)، سریعترین الگوریتم شناخته شده غربال میدان اعداد عمومی (GNFS) است. پیچیدگی زمانی الگوریتم GNFS برای فاکتورگیری یک عدد صحیح N برابر است با:

$$O(e^{(1.923+O(1)).(logN)^{1/3}.(loglogN)^{2/3}})$$

که در آن c ثابتی در حدود ۹۲۳.۱ است، که در آن N عددی است که باید فاکتورگیری شود، و عبارت o(۱) عبارت‌های مرتبه پایین‌تری را نشان می‌دهد که با بزرگ‌تر شدن N ناچیز می‌شوند. به عبارت ساده تر، پیچیدگی زمانی الگوریتم GNFS در

$$lnN^{1/3}$$

نمایی است، که در آن N ln لگاریتم طبیعی N است. این بدان معنی است که زمان اجرای الگوریتم به صورت نمایی با ریشه مکعب رشد می کند. از تعداد ارقام در .N این آن را به یک الگوریتم زیر نمایی تبدیل می‌کند که به طور قابل‌توجهی سریع‌تر از الگوریتم‌های صرفاً نمایی مانند تقسیم آزمایشی یا الگوریتم rho پولارد است. با این حال، GNFS هنوز از نظر محاسباتی فشرده است و برای اعداد بزرگتر کارایی کمتری دارد. برای اعداد با اندازه متوسط تر (تا چند صد رقم)، الگوریتم های دیگر مانند غربال درجه دوم (QS) یا روش منحنی بیضی (ECM) ممکن است در عمل سریعتر باشند، علیرغم داشتن پیچیدگی های زمانی مجانبی بدتر.

پیچیدگی زمانی الگوریتم غربال درجه دوم:

$$exp((1+O(1))(lnN)^{1/2}(lnlnN)^{1/2}$$

این هنوز نمایی است، اما با نمایی کوچکتر از الگوریتم ،GNFS که آن را برای اعداد با ارقام کمتر کاربردی تر می کند. روش منحنی بیضی (ECM) دارای پیچیدگی زمانی است که به صورت نمایی در جذر کوچکترین عامل اول N است. در حالی که این روش در بدترین حالت به خوبی الگوریتم های GNFS یا QS نیست، ECM می تواند برای یافتن بسیار موثر باشد. عوامل اول کوچک اعداد بزرگ شایان ذکر است که این پیچیدگی های زمانی برای بدترین سناریوها است. در عمل، زمان اجرای واقعی بسته به عدد ورودی خاص و جزئیات پیاده سازی الگوریتم ها می تواند به طور قابل توجهی متفاوت باشد.

به طور خلاصه، الگوریتم غربال میدان اعداد عمومی (GNFS) دارای بهترین پیچیدگی زمانی مجانبی شناخته شده برای فاکتورگیری اعداد صحیح بسیار بزرگ است، با زمان اجرا نمایی در

$$lnN^{1/3}$$

که در آن N عدد فاکتورگیری شده است. با این حال، برای ورودی های کوچکتر، الگوریتم های دیگر مانند غربال درجه دوم یا روش منحنی بیضی ممکن است در عمل کارآمدتر باشند. توجه به این نکته مهم است که اینها تنها برخی از الگوریتم های شناخته شده هستند و این احتمال وجود دارد که الگوریتم سریع تری در آینده کشف شود. با این حال، هیچ الگوریتم کارآمد (زمان چند جمله ای) برای فاکتورسازی اعداد صحیح در کامپیوترهای کلاسیک وجود ندارد.

از سوی دیگر، کامپیوترهای کوانتومی این پتانسیل را دارند که اعداد صحیح بزرگ را بسیار سریعتر فاکتور کنند. الگوریتم شور، یک الگوریتم کوانتومی، می تواند یک عدد صحیح n بیتی را در چند جمله ای زمانی در n فاکتور کند. این یک تهدید قابل توجه برای سیستم های رمزنگاری فعلی است که بر دشواری

فاکتورسازی اعداد صحیح متکی هستند. محققان فعالانه روی توسعه الگوریتم‌های رمزنگاری پس کوانتومی کار می‌کنند که حتی در برابر رایانه‌های کوانتومی ایمن باشند.

۲ اعداد Semiprime چیست؟ تجزیه این اعداد نسبت به سایر اعداد پیچیده‌تر است یا ساده‌تر؟

اعداد نیمه اول که با نام اعداد نیمه اول یا اعداد دو اول نیز شناخته می شوند، اعداد طبیعی هستند که دقیقا حاصل ضرب دو عدد اول مجزا هستند. به عبارت دیگر، یک عدد نیمه اول را می توان به صورت ضرب دو عدد اول مختلف بیان کرد. به عنوان مثال، اعداد ۶، ۱۰، ۱۴، ۱۵، ۲۱، ۲۲، ۲۶، ۳۳، ۳۴، ۳۵، ۳۸، ۳۹، ۵۱، ۵۵، ۵۷، ۵۸، ۶۲، ۶۵، ۶۹، ۷۷، ۸۵، ۸۷ و ۹۱ همه اعداد نیمه اول هستند زیرا می توان آنها را به عنوان حاصل ضرب دو عدد اول مجزا بیان کرد:

- $3 \times 2 = 6$
- $5 \times 2 = 10$
- $7 \times 2 = 14$
- $5 \times 3 = 15$
- $7 \times 3 = 21$
- ...

در مورد تجزیه اعداد نیمه اول به عوامل اول، به طور کلی ساده تر از تجزیه اعداد ترکیبی که نیمه اول نیستند در نظر گرفته می شود. در اینجا دلیل آن است:

۱. تعداد محدود عامل اول: طبق تعریف، اعداد نیمه اول دقیقاً دو عامل اول مجزا دارند. این محدودیت فرآیند فاکتورسازی را در مقایسه با اعداد ترکیبی که ممکن است بیش از دو عامل اول داشته باشند، ساده می کند.

۲. الگوریتم های فاکتورسازی کارآمد: الگوریتم های کارآمدی وجود دارند که به طور خاص برای فاکتورگیری اعداد نیمه اول طراحی شده اند، مانند غربال درجه دوم و غربال فیلد اعداد. این الگوریتم‌ها از ساختار اعداد نیمه اول بهره می‌برند و فرآیند فاکتورسازی را کارآمدتر از اعداد مرکب عمومی می‌کنند.

۳. مبنای عامل کوچکتر: در زمینه الگوریتم های فاکتورسازی، پایه عامل (مجموعه اعداد اول کوچک مورد استفاده در الگوریتم) برای اعداد نیمه اول کوچکتر است زیرا عوامل به دو عدد اول محدود می شوند. این امر محاسبات را در مقایسه با اعداد مرکب با چندین عامل اول قابل کنترل تر می کند.

در اینجا یک قیاس وجود دارد: تصور کنید یک کیسه با چند تیله رنگی دارید. یافتن یک سنگ مرمر رنگی خاص (ضریب اصلی) در صورتی که سنگ مرمرهای زیادی وجود داشته باشد (عدد مرکب با فاکتورهای زیاد) می تواند دشوار باشد. اما، اگر کیسه فقط دو تیله داشته باشد (نیمه پرایم)، پیدا کردن یک سنگ مرمر رنگی خاص بسیار آسان تر می شود. با این حال، توجه به این نکته مهم است که در حالی که فاکتورسازی اعداد نیمه اول به طور کلی ساده تر از اعداد مرکب عمومی است، مشکل همچنان با افزایش اندازه (تعداد ارقام) اعداد نیمه اول افزایش می یابد. اعداد نیمه اول بزرگ هنوز هم می توانند چالش های محاسباتی قابل توجهی برای الگوریتم های فاکتورسازی ایجاد کنند. معمولاً تجزیه اعداد نیمه اول بسیار بزرگ به ضرایب اول دشوارتر از فاکتورگیری اعدادی است که اول هستند یا دارای عوامل اول کوچک هستند. این به این دلیل است که اعداد نیمه اول به طور خاص به عنوان حاصل ضرب دو عدد اول بزرگ انتخاب می شوند و هیچ الگوریتم کارآمد شناخته شده ای وجود ندارد که آنها را به طور کلی عامل بندی کند.

به عبارت دیگر، تجزیه اعداد نیمه اول بسیار بزرگ زمانی که دو عدد اول بسیار بزرگ و تصادفی انتخاب شده و دارای مقدار نسبتا نزدیک هستند، حتی سریع ترین الگوریتم ها در سریع ترین کامپیوترها آنقدر زمان می برند که در واقع ناکارآمد هستند. به طور خلاصه، اعداد نیمه اول، اعداد مرکبی هستند که حاصل ضرب

دو عدد اول مجزا هستند، و تجزیه آنها به ضرایب اول آنها معمولا ساده‌تر از تجزیه اعداد ترکیبی که نیمه اول نیستند، به دلیل تعداد محدود عوامل اول و در دسترس بودن کارآمد است. الگوریتم های فاکتورسازی که به طور خاص برای اعداد نیمه اول طراحی شده اند. اما با افزایش اندازه اعداد اول، فاکتورگیری اعداد نیمه اول به طور تصاعدی دشوارتر می شود و به طور فزاینده ای به چالش کشیدن آنها تبدیل می شود. این دشواری در طرح‌های رمزنگاری مختلف، مانند ،RSA که بر مشکل فاکتورگیری اعداد نیمه‌اول بزرگ برای امنیت خود متکی هستند، استفاده می‌شود.