دانشکده‌ی علوم ریاضی

آنالیز الگوریتم‌ها — ۲۴ اردیبهشت ۹۲

جلسه‌ی ۲۱: الگوریتم‌های تصادفی

مدرّس: دکتر شهرام خزائی — *نگارنده: سوزان اصغری*

۱ مقدمه

در این جلسه به الگوریتم‌های تصادفی می‌پردازیم. الگوریتم‌های تصادفی، الگوریتم‌هایی هستند که در حین اجرای الگوریتم از مقادیر تصادفی استفاده می‌کنند. مزیت این الگوریتم‌ها ساده بودن و یا سریع بودنشان برای حل بسیاری از مسائل است. این الگوریتم‌ها در بسیار ی از حوزه‌ها از جمله الگوریتم‌های نظریه اعداد، ساختمان داده‌ها، اتحادهای جبری، الگوریتم‌های گراف و ... به‌کار می‌روند. در ادامه انواع الگوریتم‌های تصادفی و چند مثال بیان می‌کنیم.

۲ انواع الگوریتم‌های تصادفی

الگوریتم‌های تصادفی دو نوع اند:

- الگوریتم‌هایی که جواب آن‌ها با احتمالی درست است و می‌توانیم با تکرار الگوریتم احتمال صحت آن را افزایش دهیم.
- الگوریتم‌هایی که زمان اجرای آن‌ها تصادفی است.

۳ آزمون اول بودن عدد

برای تشخیص اول بودن یک عدد، الگوریتمی معین و چندجمله‌ای با درجه‌ی بالا به نام الگوریتم ASK[۱] وجود دارد که در سال ۲۰۰۲ منتشر شده‌است و زمان اجرای آن $O(\log^{12} n)$ است اما در سال ۲۰۰۵ توسط C. Pomerance و H. W. Lenstra به الگوریتمی با پیچیدگی $O(\log^{6} n)$ بهبود یافت. اکنون به الگوریتمی تصادفی و سریع‌تر برای حل این مساله می‌پردازیم:

قضیه ۱ (*قضیه‌ی فرما*) *اگر p اول باشد و a نسبت به p اول باشد، آنگاه* $a^{p-1} = 1 \mod p$.

اما عکس این قضیه در حالت کلی برقرار نیست.

تعریف ۱ (*عدد کارمایکل*[۲]) *عدد مرکب مثبت N را کارمایکل گوییم هرگاه به ازای هر عدد a که نسبت به N اول است داشته‌باشیم* $a^{N-1} = 1 \mod N$.

لم ۲ *اگر رابطه‌ی $a^{N-1} \neq 1 \mod N$ برای یک عدد a کوچکتر از N که نسبت به N اول است برقرار باشد، آنگاه این ویژگی حداقل برای نصف اعداد برقرار است.*

[۱] Agrawal–Kayal–Saxena
[۲] Carmichael

اکنون الگوریتم تصادفی تشخیص اعداد اول را بیان می‌کنیم:

۱.۳ پیاده‌سازی

Algorithm 1 Algorithm: PRIMALITY-TEST

```
function PRIMALITY-TEST(number N)
a ← {2, ..., N − 1} ; i.e. choose a number uniformly at random

    if gcd(a, N) ≠ 1 or a^(N−1) ≠ 1 mod N then
        return NO
    else
        return YES
```

بر اساس ورودی الگوریتم سه حالت ممکن است پیش بیاید:

- ورودی اول باشد که در این صورت خروجی الگوریتم YES است.
- ورودی مرکب غیر کارمایکل باشد، در این صورت با احتمال حداقل ۱/۲ خروجی NO (خروجی صحیح) است.
- ورودی عدد کارمایکل باشد که با احتمال خیلی کمی ($\text{۱}/\phi(N)$) خروجی NO (خروجی صحیح) است.

اگر الگوریتم فوق را روی یک عدد مرکب غیر کارمایکل t بار اجرا کنیم احتمال اینکه جواب الگوریتم NO باشد بیشتر از $\text{۱} - (\text{۱}/\text{۲})^t$ است که با انتخاب مناسب t می‌توان آن را به اندازه‌ی دلخواه به ۱ نزدیک کرد.

نکته ۱ *آزمون دیگری به نام آزمون میلر- رابین وجود دارد که اگر عدد انتخاب شده اول باشد خروجی همواره YES است و اگر مرکب باشد با احتمال حداکثر ۲^{-t} (مستقل از کارمایکل بودن یا نبودن) خروجی YES است که t تعداد دفعات تکرار است.*

۴ الگوریتم تولید اعداد اول

۱.۴ پیاده‌سازی

Algorithm 2 Algorithm: GENERATE-PRIME

```
function GENERATE-PRIME(length n)
p' ← {0, 1}^(n−1)
p := 1 || p'

    if PRIMALITY-TEST(p) = YES then
        return p
```

۲.۴ احتمال موفقیت الگوریتم

قضیه ۳ *(قضیه اعداد اول لاگرانژ) تعداد اعداد اول کوچکتر از x را با $\pi(x)$ نشان دهید. در این صورت $\pi(x) \approx x/\ln(x)$، یا به طور دقیق‌تر*

$$\lim_{x\to\infty} \frac{\pi(x)}{x/\ln(x)} = \text{۱}$$

کران پایین زیر نیز توسط چبیشف[۳] برای تعداد اعداد اول کوچکتر از x داده شده است:

$$\pi(x) \geq \frac{x}{۲ \log_۲ x}$$

پس احتمال اینکه عدد تصادفی n-بیتی اول باشد، حداقل $۱/۲n$ است. لذا الگوریتم فوق را باید حدود $۲n$ بار اجرا کرد تا یک عدد اول تولید کند. به طور دقیق‌تر اگر الگوریتم $۲cn$ بار اجرا شود احتمال اینکه در هیچ‌یک از مراحل عدد اولی تولید نشود حداکثر برابراست با

$$(۱ - ۱/۲n)^{۲cn} \leq e^{-c}$$

[۳] Chebyshev