تمرین محاسبات کوانتمی

درس طراحی الگوریتم

مهسا سادات رضوی

سید صالح اعتمادی

دانشگاه علم و صنعت ۹۸-۹۷

لطفا به نکات زیر توجه کنید:

- مهلت ارسال این تمرین شنبه ۲۲ تیر ساعت ۱۱:۵۹ ب.ظ است.
- نام شاخه، پوشه و پول ریکوست همگی دقیقا "AQ" باشد.
- اگر در حل تمرین محاسبات کوانتمی مشکلی داشتید یا به اطلاعات بیشتر نیاز پیدا کردید،می‌توانید با آیدی تلگرام mhsarzvi@ تماس بگیرید.

موفق باشید.

توضیحات کلی تمرین

تمرین این هفته ی، ۶ سوال دارد که باید به همه ی این سوال ها پاسخ دهید. برای حل این سری از تمرین ها مراحل زیر را انجام دهید:

۱. ابتدا مانند تمرین های قبل، یک پروژه به نام AQ بسازید.

۲. برای استفاده از نوع دادهای BigInteger کتابخانه `System.Numerics` را به رفرنس‌ها اضافه کنید.

۳. کلاس هر سوال را به پروژه ی خود اضافه کنید و در قسمت مربوطه کد خود را بنویسید. سوال اول و دوم داده تست نداشته و با یک یونیت تست ساده، تست می‌شوند. برای سوال ششم نیاز به ایجاد یک پروژه جداگانه Q#ً است. برای توضیح بیشتر به سوال ششم مراجعه کنید.

هر کلاس شامل دو متد اصلی است:

- متد اول: تابع solve است که شما باید الگوریتم خود را برای حل سوال در این متد پیاده سازی کنید.
- متد دوم: تابع process است که مانند تمرین های قبلی در TestCommon پیاده سازی شده است.

بنابراین با خیال راحت سوال را حل کنید و نگران تابع process نباشید! زیرا تمامی پیاده سازی ها برای شما انجام شده است و نیازی نیست که شما کدی برای آن بزنید.

۴. اگر برای حل سوالی نیاز به تابع های کمکی دارید؛ می توانید در کلاس مربوط به همان سوال تابع تان را اضافه کنید.

اکنون که پیاده سازی شما به پایان رسیده است، نوبت به تست برنامه می رسد. مراحل زیر را انجام دهید.

۱. یک UnitTest برای پروژه ی خود بسازید.

۲. فولدر TestData که در ضمیمه همین فایل قرار دارد را به پروژه ی تست خود اضافه کنید.

۳. فایل‌های `GradedTests.cs` ، `Q1RSAKeysTests.cs` ، `Q2EncryptionTests.cs` را به پروژه ی تستی که ساخته اید اضافه کنید. توجه کنید که بر خلاف تمرین‌های قبل برای هر سوال یک تست در نظر گرفته شده تا Timeout هر تست جداگانه محاسبه شود و در صورت حل نکردن هر تست بتوانید برای آن تست جداگانه Assert.Inconclusive بنویسید.

دقت کنید که TestCommon تغییر یافته است. بنابراین شما باید نسخه ی جدید آن را با دستورPull git دریافت کنید .

هدف تمرین

هدف از جلسه محاسبات کوانتومی آشنایی شما با مفهوم و ایده کلی الگوریتم‌های کوانتومی بود. با وجود بعضی مفاهیم تئوری و نسبتا پیچیده، فکر می‌کنم اکثرا با ایده اصلی آشنا شدید. ایده اصلی رابطه محاسبه و آزمایش است. مثلا برای پیدا کردن جواب 2×2 می‌توان یک آزمایش انجام داد، به این ترتیب که با سرعت دو کیلومتر در ساعت و به مدت دو ساعت راه بروید. بعد اندازه بگیرید که چقدر راه رفته‌اید. این می‌شود جواب محاسبه. پس برای محاسبه 2×2 می‌توان یک آزمایش ترتیب داد و جواب را از این طریق پیدا کرد. بدون وارد شدن به جزئیات قبول می‌کنیم که برخی محاسبات کوانتومی بسیار پیچیده بوده و محاسبه جواب، مستلزم انجام محاسبات بسیار زیاد است (از مرتبه توانی نسبت به اندازه ورودی). اما انجام آزمایش و اندازه گیری جواب را می‌توان در کسری از ثانیه انجام داد. حال چنانچه بتوان محاسبات لازم برای یک مساله پیچیده محاسباتی (مثلا تجزیه یک عدد بزرگ به عوامل اول آن) را متناظر با محاسبات لازم برای یک (سری) آزمایش‌های کوانتومی کرد، می‌توان در عمل آزمایش کوانتومی را انجام داد و نتیجه آن را اندازه گیری کرد. به عبارت دیگر، یک سری آزمایش کوانتومی را طوری طراحی می‌کنیم که نتیجه آزمایش متناظر با جواب مساله محاسباتی ما باشد. پس به این ترتیب می‌توان با استفاده از تناظر محاسبات و آزمایش فیزیکی، مسائل محاسباتی پیچیده را حل کرد.

یکی از دانشمندان [1] علوم کامپیوتر می‌گوید: «تا کدش رو نزنم، درست متوجه نمیشم». هدف ما در این تمرین جا افتادن دو مفهوم/مطلب است.

۱. اهمیت کامپیوتر کوانتومی در قالب یک کاربرد بسیار مهم و واقعی: الگوریتم رمزنگاری RSA پایه و اساس اکثر ارتباطات امن بانکی و نظامی است. لازمه شکستن این رمز تجزیه یک عدد اول بزرگ به عوامل اولش است. این یک مساله NP-Complete است که حل آن برای ورودی به اندازه کافی بزرگ روی یک ابر کامپیوتر میلیاردها سال طول خواهد کشید. اما یک کامپیوتر کوانتومی به اندازه کافی بزرگ می‌تواند با الگوریتم Shor این کار را در مدت کوتاهی انجام دهد. هدف تمرین‌های ۱ تا ۵ آشنایی شما با الگوریتم RSA و شکستن رمز آن با کامپیوتر معمولی و راه حل کلی برای پیاده‌سازی الگوریتم Shor می‌باشد. با توجه به پیچیدگی الگوریتم Shor و میزان انتظار ما از تسلط شما به الگوریتم‌های کوانتومی، در این تمرین‌ها الگوریتم Shor را روی کامپیوتر معمولی پیاده‌سازی خواهیم کرد.

[1] اسمش رو پیدا نکردم. اگر شما پیدا کردین به من هم بگین.

۲. روش و چگونگی ارتباط پیچیدگی محاسباتی توانی با آزمایش کوانتومی: پس از آشنایی با شیوه استفاده از کامپیوتر کوانتومی برای یک کاربرد واقعی، تلاش تمرین شماره ۶، آشنایی شما با ساده‌ترین الگوریتم کوانتومی، Deutsch–Jozsa ، و شیوه پیاده‌سازی آن در زبان Q# و کامپیوتر کوانتومی IBM است.

۱ بدست آوردن کلید خصوصی و عمومی رمزگذاری RSA

در مرحله اول لازم است با الگوریتم رمزنگاری RSA آشنا شویم. الگوریتم رمزنگاری RSA یک الگوریتم رمزگذاری نامتقارن [۲] است که با استفاده از جفت کلید عمومی و خصوصی متن را encode و decode می‌کند. برای فرستادن پیام خصوصی برای یک نفر، عمل encoding را با کلید عمومی آن فرد انجام داده و برای او می‌فرستیم. گیرنده پیام در مقابل عمل decoding را توسط کلید خصوصی متناظر با آن کلید عمومی انجام داده و به متن پیام دسترسی پیدا می‌کند. در این سوال با شیوه تولید کلیدهای عمومی و خصوصی آشنا می‌شویم. طریقه ساختن این کلیدها به صورت زیر است:

- دو عدد اول p و q را انتخاب میکنیم.
- اگر $p \times q = n$ و $(p-1) \times (q-1) = f$ باشد آنگاه عدد e را طوری انتخاب میکنیم که از n کوچکتر بوده و نسبت به f اول باشد.
- جفت اعداد n و e کلید عمومی رمزگذاری خواهند بود.
- کلید خصوصی d به صورت مقابل محاسبه میشود.

$$d \times e \equiv 1 \pmod{f}$$

بنابراین d عددی است که باقی مانده حاصل ضرب d در e بر f برابر با 1 شود.

در این سوال شما باید با استفاده از سه عدد p و q و e که در ورودی به شما داده شده است کلید عمومی و کلید خصوصی رمزگذاری را محاسبه کنید. در سه خط اول فایل ورودی به ترتیب p و q و e وجود دارد. در خط اول فایل خروجی کلید خصوصی یعنی d را برگردانید.

– راهنمایی: برای بررسی درستی جواب ها می توانید از این لینک استفاده کنید.

ورودی نمونه ۱ :

[۲] الگوریتم‌های رمزنگاری نامتقارن دارای دو کلید می‌باشند. متنی که با یکی از کلیدها رمز شود، با کلید دیگر رمزگشایی می‌شود. در مقابل آن الگوریتم رمزنگاری متقارن است که دارای یک کلید است. در این الگوریتم، هر متنی با همان کلیدی که رمز شده، رمزگشایی می‌شود.

```
19
11
7
```

خروجی نمونه ۱ :

```
103
```

ورودی نمونه ۲ :

```
7
11
17
```

خروجی نمونه ۲ :

```
53
```

ورودی نمونه ۳ :

```
11
13
23
```

خروجی نمونه ۳ :

```
47
```

۲ رمزگذاری با استفاده از کلید عمومی

بعد از تسلط به تولید کلیدهای عمومی و خصوصی، نوبت استفاده از کلید عمومی برای رمزنگاری نامتقارن یک پیام است. همانطور که در سوال قبل گفته شد در الگوریتم RSA رمز کردن متن به کمک کلید عمومی صورت میگیرد. در این سوال شما باید با داشتن کلید عمومی متن داده شده را رمز کنید. اگر n و e را به عنوان کلید

عمومی در اختیار داشته باشیم و m متن مورد نظر ما برای رمز کردن باشد c یعنی کاراکتر رمز شده متناظر با m به شکل زیر محاسبه می شود:

$$c = m^e \bmod n$$

در خط اول و دوم فایل ورودی به ترتیب کلیدهای n و e داده شده اند. سپس در خط بعدی متنی که باید با الگوریتم RSA رمز کنید آمده است. در فایل خروجی رمز شده متن ورودی با فرمت نشان داده شده در زیر قرار می گیرد. این فرمت بصورت آرایه ای از اعداد است که هر یک نشان دهنده معادل رمز شده ی یک کاراکتر است. این اعداد با ’،’ از یکدیگر جدا شده اند.

ورودی نمونه ۱ :

```
253
23
salam
```

خروجی نمونه ۱ :

```
92,212,223,212,109
```

ورودی نمونه ۲ :

```
253
7
algorithm
```

خروجی نمونه ۲ :

```
224,202,214,199,137,239,162,223,43
```

ورودی نمونه ۳ :

```
187
7
abcdef
```

خروجی نمونه ۳ :

```
92,21,176,144,84,119
```

۳ پیدا کردن کلید رمزنگاری با بدست آوردن عامل های اول اعداد

پس از آشنایی با تولید کلیدهای عمومی/خصوصی و رمزنگاری یک پیام توسط کلید عمومی، نوبت به رمزگشایی از یک متن رمز شده بدون در دست داشتن کلید رمز آن می‌باشد. شکستن رمز RSA در صورتی که بتوانیم عامل های اعداد خیلی بزرگ را بدست آوریم امکان پذیر می شود. پیدا کردن اعداد اول بزرگ و ضرب کردن آنها برای بوجود آمدن یک کلید رمز کار نسبتا ساده‌ای است اما برعکس آن یعنی بدست آوردن فاکتورهای اول اعداد خیلی بزرگ برای شکستن کد ساده نیست.

همانطور که در سوال های قبل نحوه ساخته شدن کلید خصوصی در الگوریتم RSA را دیدید اگر بتوانیم با داشتن کلید عمومی و بدست آوردن فاکتورهای عدد n یعنی p و q میتوانیم کلید خصوصی رمزگذاری را بیابیم و در نتیجه متن رمز شده را رمزگشایی کنیم. در این سوال کلید عمومی رمزگذاری یعنی n و e به شما داده می شود و شما باید با Factorize کردن n و بدست آوردن p و q کلید خصوصی را بدست آورید.

در خط اول و دوم فایل ورودی n و e قرار دارد و در فایل خروجی کلید خصوصی (d) برگردانده می شود.

ورودی نمونه ۱ :

```
209
7
```

خروجی نمونه ۱ :

```
103
```

ورودی نمونه ۲ :

```
391
13
```

خروجی نمونه ۲ :

```
325
```

ورودی نمونه ۳ :

```
517
17
```

خروجی نمونه ۳ :

```
433
```

۴ پیدا کردن عامل های اول کلید عمومی با استفاده از الگوریتم Shor

در الگوریتم Shor پریود یک عدد بزرگ را با کوانتوم کامپیوتر بدست می‌آوریم. بعد از بدست آوردن پریود، با محاسبات ساده‌ای می‌توان عوامل اول را پیدا کرد. در این تمرین تمام مراحل الگوریتم Shor را روی یک کامپیوتر معمولی انجام می‌دهیم با این هدف که شما متوجه بشوید، کدام قسمت از الگوریتم وقت گیر است و با کامپیوتر کوانتومی در زمان چند جمله‌ای انجام می‌شود.
فرض کنید N عددی است که به دنبال یافتن عامل های اول آن هستیم. (برای سادگی تنها دو عامل اول p و q را برای n در نظر میگیریم بطوریکه $p \times q = N$) برای پیدا کردن عامل های N در یک راه حل Force Brute تک تک اعداد اول را بر N تقسیم کرده و باقی مانده را با x بصورت زیر نشان می دهیم.

$$a \equiv x \pmod{N}$$

اگر a را به توان برسانیم و باقی مانده اعداد بدست آمده بر N را بررسی کنیم می‌بینیم که با تکرار عمل به توان رساندن باقی مانده ها با پریود مشخصی در یک الگو تکرار می شوند. برای مثال a را برابر با ۲ و N را برابر با ۷ درنظر بگیرید.
در این مثال پریود برابر با ۳ است. دقت کنید که آخرین باقی مانده در الگو تکرار شونده همیشه برابر با ۱ است.
در یک تعریف کلی اگر پریود $N \pmod{x}$ را r بنامیم، r برابر است با کوچکترین عددی که در شرط زیر صدق کند:

$$x^r \equiv 1 \pmod{N}$$

حال برای بدست آوردن p و q بصورت الگوریتم زیر عمل می کنیم.

۱. در قدم اول a را بصورت رندوم طوری انتخاب می کنیم که کوچکتر از N باشد و نسبت به آن اول باشد.

۲. در قدم دوم r یعنی پریود $N \pmod{x}$ را حساب میکنیم.

۳. در قدم سوم بررسی کنید که r زوج باشد و در شرط زیر صدق کند.

$$x^{r/2} + 1 \neq 0 \pmod{N}$$

۴. در قدم آخر p و q بصورت زیر قابل محاسبه اند.

$$\Longrightarrow p = gcd(a^{r/2} - 1, N), q = gcd(a^{r/2} + 1, N)$$

در این سوال باید با داشتن کلید عمومی یعنی n فاکتورهای اول آن یعنی p و q و همچنین پریود r را محاسبه کنید. در فایل ورودی یک عدد قرار دارد که باید آن را به عامل های اول آن تجزیه کنید. در فایل خروجی فاکتورهای عدد داده شده را به ترتیب صعودی و هر یک را در یک خط چاپ کنید. سپس در خط بعد آخرین عددی که به عنوان پریود قابل قبول بدست آورده اید را چاپ کنید. اگر قبل از پیدا کردن پریود فاکتورها را بصورت رندوم پیدا کردید در خط سوم ۱- را چاپ کنید.

نکته : برای تولید اعداد رندوم از $seed = 0$ استفاده کنید.

ورودی نمونه ۱ :

```
145
```

خروجی نمونه ۱ :

```
5
29
14
```

ورودی نمونه ۲ :

```
407
```

خروجی نمونه ۲ :

```
11
37
10
```

۵ شکستن رمز RSA با پیدا کردن فاکتورهای اول

دراین سوال با کنار هم گذاشتن مسائل قسمت‌های قبل میخواهیم تنها با داشتن کلید عمومی رمزگذاری، متن رمز شده را بشکنیم و به متن اصلی دست یابیم. برای این کار لازم است که بتوانیم از روی کلید عمومی فاکتورهای

اول را بدست اوریم.
در خط اول و دوم فایل ورودی کلید عمومی یعنی n و e داده شده است و در خط بعدی آن متنی رمز شده بصورت آرایه ای از اعداد صحیح قرار دارد. شما باید در فایل خروجی رشته متن اصلی (Plain text) را برگردانید.

ورودی نمونه ۱ :

```
253
23
92,212,223,212,109
```

خروجی نمونه ۱ :

```
salam
```

ورودی نمونه ۲ :

```
187
7
92,21,176,144,84,119
```

خروجی نمونه ۲ :

```
abcdef
```

۶ الگوریتم کوانتومی Deutsch-Jozsa

هدف از این سوال آشنایی با زبان `Q#` و پیاده سازی یک الگوریتم ساده روی کامپیوتر کوانتومی است. کامپیوتر کوانتومی IBM به ما این امکان را می دهد تا بتوانیم برنامه کوانتومی خود را بصورت آنلاین روی آن اجرا کنیم.

۱.۶ آماده‌سازی

ابتدا لازم است Quantum Development Kit را برای ویژوال استودیو نصب کنید. سپس یک پروژه از نوع `Q# Application` داخل پوشه AQ که در ریشه گیت قرار دارد، به نام `DJ` درست کنید.

۲.۶ آشنایی با زبان Q#

برای شروع یک برنامه با زبان Q# نوشته و آن را اجرا کنید. در این لینک نحوه اجرای یک برنامه کوانتومی (برای مثال ایجاد یک حالت superposition) توضیح داده شده است. علاوه بر منبع ذکر شده، می‌توانید از منابع زیر نیز برای آشنایی با زبان Q# استفاده کنید.

- Udemy QC101
- Q# introduction from Art Of Engineering
- Q# introduction from The New Stack
- Q# introduction blog from Microsoft
- Q# introduction from c-sharpcorner
- Q# community website
- Udemy Quantum Computing overview course

۳.۶ آشنایی با الگوریتم Deutsch-Jozsa

الگوریتم Deutsch-Jozsa یک الگوریتم ساده است که اجرای آن روی کامپیوتر کلاسیک با پیچیدگی نمایی امکان پذیر است. اگر بتوانیم آن را روی کامپیوتر کوانتومی اجرا کنیم میتوانیم در زمان خطی آن را حل کنیم. فرض کنید تابعی به نام f داریم که یک بیت 0 یا 1 بعنوان ورودی گرفته و یک بیت 0 یا 1 بعنوان خروجی میدهد. اگر این تابع به ازای هر مقدار ورودی (0 یا 1) همواره یک خروجی دهد آن را constant می نامیم و اگر خروجی آن به ازای ورودی های مختلف بین 0 و 1 تغییر کند آن را balanced می نامیم. هدف از الگوریتم Deutsch-Jozsa این است که تشخیص دهد تابع f از کدامیک از انواع گفته شده است. اگر بخواهیم برای ورودی تک بیتی این سوال را بصورت کلاسیک حل کنیم نیاز است که ۲ بار با دادن ۲ حالت مختلف ورودی (۰ یا ۱) خروجی را بسنجیم. اگر ورودی را ۲ بیتی در نظر بگیریم لازم است ۴ بار خروجی تابع را بسنجیم . در راه حل کوانتومی میتوانیم با در نظر گرفتن حالات مختلف ورودی بصورت superposition ای از qbit ها آن را در زمان خطی حل کنیم.

برای حل این سوال ابتدا الگوریتم خود را با کمک این لینک به زبان Q# پیاده سازی کرده و بعد اضافه کردن کامنت‌هایی که نشان از فهم الگوریتم و زبان Q# باشد پروژه را در گیت اضافه و پوش کنید. سپس در سرویس محاسبات کوانتومی شرکت IBM در این آدرس، حساب کاربری باز کرده و الگوریتم خود را توسط بخش Circuit Composer طراحی کرده و روی کامپیوتر کوانتومی ibmq_16_melbourne اجرا

کنید. از مدار، مراحل و نتیجه آن `screen-shot` گرفته و در گزارشی به نام Report.pdf در ریشه پروژه DJ بگنجانید. در گزارش علاوه بر نتیجه آزمایش، به مراحل پیاده‌سازی الگوریتم روی کامپیوتر IBM نیز اشاره کنید.

راهنمایی : برای آشنا شدن با نحوه عملکرد گیت های مورد استفاده برای پیاده سازی الگوریتم از این لینک استفاده کنید.