Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Számítástudományi és Információelméleti Tanszék

Függvény- és gráftulajdonságok lokális tesztelése

Diplomaterv

Készítette
Szabó Dániel

Konzulens
dr. Friedl Katalin

2021. május 21.

Tartalomjegyzék

Kivonat **i**

Abstract **ii**

1. Bevezetés **1**
1.1. A PCP-elmélet és a tulajdonságtesztelés rövid története 2
1.2. Különböző PCP formalizmusok 4

2. Tulajdonságtesztelés **6**
2.1. Ígéretproblémák (promise problems) 6
2.1.1. Néhány definíció 7
2.1.2. Hézagproblémák (gap problems) 9
2.2. Tulajdonságtesztelés 9
2.2.1. Alapvető definíciók 9
2.2.2. A távolságparaméter szerepe 11

3. Függvénytulajdonság-tesztelés **13**
3.1. Linearitás teszt 13
3.1.1. A tesztelő algoritmus 14
3.1.2. A témakör fő tétele 14
3.2. Alacsonyfokúság tesztelése 17
3.2.1. Egyváltozós eset 17
3.2.2. Többváltozós eset 18
3.3. Diktatúra és junta 19

4. Gráftulajdonság-tesztelés **21**
4.1. Sűrűgráf-modell 22
4.1.1. Néhány egyszerűen tesztelhető gráftulajdonság 22
4.1.2. A k-színezhetőség tesztelése 23
4.1.3. Gráfok párosságának tesztelése 24
4.2. Egyéb gráfmodellek 27
4.2.1. Korlátos fokszámú gráfok modellje 27
4.2.2. Általános gráfmodell 29

5. Feszített-C_4-mentesség tesztelése **31**

6. Kapcsolódó kvantuminformatikai eredmények **41**
6.1. Kvantum-tulajdonságtesztelés 41
6.1.1. Amplitúdó-erősítés 42
6.1.2. Bernstein–Vazirani-algoritmus 42
6.2. Junták tesztelése 43

6.3. CHSH-játék avagy Bell-egyenlőtlenség 45
6.3.1. Klasszikus eset 45
6.3.2. Kvantumos eset 46

7. Elméleti eredmények verifikálása **48**
7.1. Linearitás teszt 48
7.2. Gráfok párosságának tesztelése 51

8. Összegzés **54**

Köszönetnyilvánítás **56**

Irodalomjegyzék **57**

HALLGATÓI NYILATKOZAT

Alulírott *Szabó Dániel*, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2021. május 21.

Szabó Dániel
hallgató

Kivonat

A PCP-elmélet (Probabilistically Checkable Proofs) alapkérdése, hogy ha adott egy állítás és egy bizonyíték, ami elvileg tanúsítja az állítás igazságát, akkor a bizonyíték végigolvasása nélkül, csak néhány helyen beletekintve, nagy valószínűséggel meg tudunk-e győződni arról, hogy valóban az állítást igazolja.

Ennek fontos része a lokális tesztelés, avagy tulajdonságtesztelés. Ez alatt azt érjük, hogy egy objektum egy tulajdonságát vizsgálva nem végzünk kimerítő ellenőrzést, hanem megelégszünk azzal, hogy az objektumnak csak egy kis részét figyelve nagy valószínűséggel helyes eredményt kapunk, jóval hatékonyabban.

Függvények vizsgálatakor például nagyon hasznos tud lenni, ha tudjuk a függvényről, hogy rendelkezik egy tulajdonsággal, például lineáris. Hasonló a helyzet gráfoknál is. Manapság rengeteg nagy hálózat vesz körül bennünket, és ezek elemzésekor fontos megtudnunk, hogy a gráf rendelkezik-e valamely tulajdonsággal. A hálózatok kiterjedt mérete miatt a kimerítő ellenőrzés sokszor nem járható út, ezért érdemes lokális tesztelést végezni.

A dolgozatban több ilyen függvény-, illetve gráftulajdonsággal kapcsolatos korábbi elméleti eredményt is bemutatunk. A függvények linearitásának tesztelését és annak elemzését részletesen ismertetjük, ezenkívül az alacsonyfokúság, valamint az ún. diktatúra és junta tulajdonságok ellenőrzésére is mutatunk algoritmust. Gráfok esetén a párosságot és azt a tulajdonságot vizsgáljuk részletesen, hogy a gráf nem tartalmaz feszített részgráfként négy hosszú kört. Ez utóbbi esetben a bizonyítás gondos végigkövetésével és pontosabb becslésekkel sikerült az eredmény konstansaiban némi javítást elérni.

A kvantuminformatika napjainkban tapasztalható jelentős fejlődése, valamint elméleti érdekessége miatt a dolgozatban a vizsgált terület néhány kvantuminformatikai vonatkozását is bemutatjuk. Konkrétan a junták tesztelésének elemzését ismertetjük, valamint a PCP-elméletnél is felmerülő interaktív bizonyító rendszerekhez kapcsolódó CHSH-játékot.

Fontos kérdés továbbá, hogy az elméleti korlátokhoz hogyan viszonyulnak a gyakorlatban tapasztalt eredmények. Ezért a függvények linearitásának és a gráfok párosságának teszteléséhez kapcsolódó algoritmusok implementálása és futtatása során kapott tapasztalati eredményeket bemutatjuk, és összevetjük az elméleti korlátokkal.

Abstract

The fundamental question of PCP theory (Probabilistically Checkable Proofs) is the following. We are given a statement and a proof that allegedly proves the statement. Is it possible to decide (correctly, with high probability) whether the statement is actually true without reading the whole proof, by only looking at a small portion of it?

Local testing or property testing plays an important role in this theory. In this setting, we are interested in a property of an object, but we do not perform an exhaustive checking, we only examine a small part of the object. This way, we can get the result much more efficiently, and it will be correct with high probability.

For example, when we examine functions, it can be very useful to know if the function has a certain property, e.g. if it is linear. The situation is similar with graphs. Nowadays, we are surrounded by many huge networks, and for analysing them it is important to know if they have a certain property. Due to the large size of the graphs, oftentimes it is not possible to check the property exhaustively, so property testing seems to be promising.

In the thesis, multiple former theoretical results are presented about function and graph properties like this. We review the testing algorithm for the linearity of functions as well as its analysis. Furthermore, algorithms for testing low-degree functions, dictatorships and juntas are also presented. In the case of graphs, we examine in detail bipartiteness and the property of not containing a cycle of length four as an induced subgraph. In the latter case, we managed to slightly improve some constants in a result by using some more accurate bounds in its proof.

Because of the quick progression and the theoretical curiosity of quantum computing, some related results of this field are also presented. Concretely, the analyses of junta-testing and the CHSH game are reviewed. The latter is related to the interactive proof systems that have a connection to PCP theory.

The relationship of the theoretical bounds and the results of real-world experiments is also an important question. That is why the algorithms for testing the linearity of functions and the bipartiteness of graphs were implemented, and the data gained by running them are presented and compared to the theoretical bounds.

1. fejezet

Bevezetés

Napjainkban gyakori probléma, hogy rengeteg adatból szeretnénk kinyerni valamilyen információt (big data). Ilyen esetben már az adat végigolvasása is rendkívül hosszadalmas lehet, ezért sokszor a hagyományos értelemben vett hatékony algoritmusok, amelyek a bemeneti adat méretében polinomiálisak, de még ha lineáris lépésszámúak is, sem elég gyorsak. Szublineáris futásidejű algoritmusra van szükség, ami nem is olvashatja végig a bemenetét.

Ezt a továbbiakban úgy értelmezzük, hogy adott egy nagy méretű objektum (ezt írja le a rengeteg adat), amelynek valamely globális tulajdonságára vagyunk kíváncsiak (azt az információt szeretnénk kinyerni, hogy rendelkezik-e az objektum ezzel a tulajdonsággal). Mindezt olyan algoritmussal kell megvalósítani, ami kevesebb lépést végez, mint az objektum mérete. Ezt nevezzük tulajdonságtesztelésnek vagy lokális tesztelésnek. Mivel globális tulajdonságról van szó, a bizonyosan korrekt válaszhoz szükség lenne az objektum végigolvasására. Tehát ilyen módon nem biztosan, csak nagy valószínűséggel tudunk helyes választ kapni, cserébe jóval hatékonyabb lehet az algoritmus.

Mint kiderül, nem tudunk jelentősen növelni a hatékonyságon, ha azt a két esetet szeretnénk megkülönböztetni, hogy teljesül-e a tulajdonság vagy nem. Valójában háromféle bemenetet kaphatunk: „teljesül rá a tulajdonság”; „távol van attól, hogy teljesüljön rá a tulajdonság”; „szürke zóna: nem teljesíti a tulajdonságot, de közel van hozzá”. A tulajdonságtesztelő algoritmusok az első két esetet különböztetik meg nagy valószínűséggel, a szürke zónába eső bemenetek ebben a kontextusban érdektelenek. Ez úgy is interpretálható, hogy a bemenet valamennyire zajos lehet, például egy traffipax csak akkor büntet, ha a mért sebesség a hibahatáron felül meghaladja a korlátot.

A bemeneti objektum lehet például egy függvény, a vizsgált tulajdonság pedig az, hogy a függvény lineáris-e: sok esetben jóval egyszerűbbé teszi egy függvénnyel való munkát, ha tudjuk róla, hogy lineáris. Ennek ellenőrzése azonban rengeteg időt vehet igénybe: adott f függvény esetén bármely három bemenetre meg kellene nézni, hogy egy egyenesre esnek-e. Ehelyett sokkal gyorsabb megoldás, ha csak néhány véletlen elemre ellenőrizzük ezt. Ugyanakkor kérdés, hogy ez a módszer mennyire megbízható eredményt szolgáltat.

Gráftulajdonságokat is lehet tesztelni, például azt, hogy a bemeneti gráf páros gráf-e. Maga a tesztelő algoritmus ez esetben is nagyon egyszerű lesz: egy véletlenszerűen kiválasztott feszített részgráfról ellenőrizzük, hogy páros-e. A téma bonyolultságát az adja, hogy be kell bizonyítani, hogy ezek az egyszerű algoritmusok valóban elég nagy valószínűséggel adnak helyes eredményt.

Egy másik tesztelhető gráftulajdonság az, hogy a bemeneti gráfban nincs négy hosszú feszített kör. Ennek a feladatnak a komplexitásáról érdekes eredmények születtek, azonban előfordul, hogy egy cikkben nem fordítanak figyelmet a végeredmény minél pontosabb meghatározására, csupán a nagyságrendjére. Bár ez a hozzáállás elméleti szemmel teljesen

érthető, az alkalmazások szempontjából már fontosak lehetnek az ezen belüli különbségek is. Ezért valamennyit javítottunk/pontosítottunk egy ilyen eredményben szereplő konstansokon.

Ha már szóba kerültek az alkalmazások, azt is fontos látni az algoritmuselméleti eredményeknél, hogy a tételekből következő elméleti korlátokhoz képest a valóságban általában milyen hatékonysággal működnek az algoritmusok. Emiatt néhány tulajdonságtesztelő algoritmust implementáltunk, hogy összevethessük a gyakorlati eredményeket az elméleti korlátokkal.

A tulajdonságteszteléshez kapcsolódik a PCP-elmélet (Probabilistically Checkable Proofs) területe. Itt egy állítás igazságáról szeretnénk meggyőződni, és ehhez rendelkezésünkre áll valamilyen bizonyíték, aminek a végignézésével elvileg meggyőződhetünk róla. Azonban ez hosszadalmas lehet, ezért szeretnénk, ha csak néhány helyen belenézve a bizonyítékba is nagy valószínűséggel helyesen tudnánk dönteni a kérdésben.

Ez hasonló a közismert **NP** nyelvosztály tanú tételéhez. Ennek értelmében azok az L nyelvek találhatók **NP**-ben, amelyekre igaz, hogy ha $x \in L$, akkor (és csak akkor) x-hez adható egy ($|x|$-ben) polinomiális méretű y tanú (avagy bizonyíték), ami polinom időben ellenőrizhető. Ez az ellenőrző algoritmus általában végignézi y-t, annak valamilyen tulajdonságait vizsgálva. A PCP-elmélet azt a lehetőséget vizsgálja, ha nem olvassuk végig a bizonyítékot (tanút), hanem csak néhány véletlenszerű helyen belenézünk. Ekkor is eldönthető nagy valószínűséggel helyesen, hogy valóban meggyőző bizonyíték-e y?

A kvantuminformatikának is vannak kapcsolódó területei. Bemutatjuk a tulajdonságtesztelés kvantumos változatának alapjait, valamint egy példát is erre. Ezt összevetjük a klasszikus megfelelőjével (amelyet szintén bemutatunk a dolgozatban). Ezenkívül az ún. interaktív bizonyító rendszereket is megemlítjük, mind klasszikus, mind kvantumos változatban.

A dolgozatot a fejezet további részében egy történeti áttekintéssel és a PCP-elmélet általános bemutatásával kezdjük. A 2. fejezetben általánosan a tulajdonságtesztelésről ejtünk szót. Ezután speciálisan a függvények lokális tesztelésével kapcsolatos néhány korábbi elméleti eredmény leírása következik a 3., valamint gráfokkal kapcsolatosak a 4. fejezetben. Egy konkrét gráftulajdonság, konkrétan a feszített-C_4-mentesség tesztelséről szól az 5. fejezet, ahol egy korábbi bizonyítást követve, de jobb becsléseket használva pontosítottunk egy eredményen. Az elméleti eredményekkel foglalkozó fejezetek közül a 6. az utolsó, ez a tárgyalt terület kvantuminformatikai vonatkozásának alapjait ismerteti. A 7. fejezetben néhány bemutatott módszer saját implementációja révén tapasztalt gyakorlati eredményeket vetjük össze az elméleti korlátokkal. A dolgozat a 8. fejezet összegzésével zárul.

1.1. A PCP-elmélet és a tulajdonságtesztelés rövid története

Az alfejezet fő forrása [Din05]. A PCP-elmélet szorosan kötődik az interaktív bizonyításokhoz (**IP** – Interactive proofs). Ezekről először az 1985-ös STOC konferencián megjelent cikkükben írt Goldwasser, Micali és Rackoff [GMR85]. Az **IP** az **NP** nyelvosztályt bővíti olyan módon, hogy egyrészt már nemcsak egy tanút kap az *ellenőr* valamilyen nagy tudású orákulumtól (*bizonyító*), hanem interakcióban van vele, azaz polinom sok kérdés-válasz üzenet mehet kettejük között. Másrészt privát módon véletlent használhat az ellenőr (könnyen belátható, hogy determinisztikus interakciók esetén pontosan **NP**-t kapjuk). Ez azt jelenti, hogy az ellenőr az üzenetei (kérdései) előtt „érméket dobálhat" úgy, hogy a kérdése függhet a kapott véletlentől, de a dobások eredményéről a bizonyító nem tud. Ha van ilyen szabályok szerinti protokoll, amelyet használva igaz egy L nyelvre, hogy $x \in L$ esetén van olyan bizonyító, amivel kommunikálva az ellenőr legalább 2/3 valószínűséggel

elfogadja x-et, és $x \notin L$ esetén bármely bizonyítóval kommunikálva legalább 2/3 valószínűséggel elutasítja azt, akkor $L \in \mathbf{IP}$. Belátható, hogy nem változik meg a nyelvosztály, ha az első korlátot 2/3-ról megnöveljük, akár 1-re, és ha a másodikat 0-hoz tetszőlegesen közelinek választjuk ($1 - 2^{-n}$-nek, ahol $n = |x|$).

Az előbb említett szerzőktől függetlenül, ugyanazon a konferencián megjelent cikkében dolgozta ki Babai az Arthur–Merlin-bizonyítások elméletét [Bab85]. Itt az **IP**-hez hasonló interaktív bizonyításokról van szó, a lényegi különbség az, hogy publikus véletlent használ az ellenőr (Arthur). Ez megfelel annak, hogy Arthur üzenetei a bizonyítónak (Merlin) csupán véletlen bitek sorozatai. Az $\mathbf{AM}[k]$ nyelvosztály azokból a nyelvekből áll, amelyek felismerhetők k db üzenetet használó protokollal, ahol az első üzenetet Arthur küldi. Belátható, hogy konstans k-ra $\mathbf{AM}[k] = \mathbf{AM}[2]$. 1986-ban Goldwasser és Sipser belátta, hogy polinom sok üzenet esetén $\mathbf{AM}[\mathrm{poly}(n)] = \mathbf{IP}$ [GS86]. Ezentúl az **IP** jelölést használjuk.

Az világos volt, hogy $\mathbf{NP} \subseteq \mathbf{IP}$, és az is, hogy $\mathbf{IP} \subseteq \mathbf{PSPACE}$. 1986-ban Goldreich, Micali és Wigderson megmutatták, hogy a gráf-nemizomorfizmus probléma – amiről nem ismert, hogy **NP**-beli lenne – benne van **IP**-ben [GMW86]. Így azt sejtették, hogy az **IP** az **NP**-nek egy enyhe kiterjesztése. 1988-ban Ben-Or, Goldwasser, Kilian és Wigderson bevezette a több-bizonyítós interaktív bizonyítások osztályát, ahol több, egymástól elszeparált bizonyítóval kommunikálhat az ellenőr: **MIP** (Multi-prover Interactive Proofs). Megmutatták, hogy polinom sok bizonyító ugyanazt a nyelvosztályt eredményezi, mint két bizonyító [BGKW88].

Mivel a legtöbben arra számítottak, hogy az **IP** csak enyhe kiterjesztése az **NP**-nek, meglepetéskét jött néhány hír 1989 végén. Először Nisan küldött egy emailt, melyben belátta, hogy $\mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{MIP}$, majd Lund, Fortnow és Karloff adott bizonyítást arra, hogy $\mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{IP}$ [LFKN90]. Ebből következik, hogy az egész polinomiális hierarchia is **IP**-ben van. Két héttel később, karácsony másnapján elküldött emailjében Shamir megmutatta, hogy $\mathbf{IP} = \mathbf{PSPACE}$ [Sha90]. Három hétre rá Babai, Fortnow és Lund azt bizonyította be, hogy $\mathbf{MIP} = \mathbf{NEXP}$ [BFL90].

Miután kiderült, hogy az **IP** ennyire tág osztály, a kutatók olyan korlátozásokat kerestek az ellenőr képességeire nézve, amelyek betartásával végül visszakapjuk **NP**-t. Volt, aki az ellenőr tárhasználatát [Con91], más az idejét [BFLS91] próbálta korlátozni. Ezekből a kísérletekből is születtek érdekes eredmények, például közelítő algoritmusok tekintetében. A leggyümölcsözőbb iránynak végül az bizonyult, ahol az ellenőr véletlen bitjeinek számát és a bizonyításból kiolvasott bitek számát egyszerre korlátozzuk: ez a PCP-k (Probabilistically Checkable Proofs) területe.

A továbbiakban a következő forgatókönyvvel dolgozunk [AB09]: az ellenőr egy V polinom időkorlátos randomizált Turing-gép (algoritmus), aki az $x \in \{0,1\}^n$ bemenet L nyelvbe tartozásáról szeretne meggyőződni. Ehhez kap egy $\pi \in \{0,1\}^*$ bizonyítást, $O(r(n))$ véletlen bitet használhat, és $O(q(n))$ bitet kérdezhet le π-ből, méghozzá adaptivitás nélkül (azaz a következő lekérdezés nem függhet a korábbiakra kapott eredménytől). Jelölje $V^\pi(x)$ azt a valószínűségi változót, hogy V elfogadja-e, hogy $x \in L$, ha a fent leírt módon hozzáférése van a π bizonyításhoz: az elfogadás eseményét $V^\pi(x) = 1$ jelöli.

1.1. Definíció. *A* $\mathbf{PCP}[r(n), q(n)]$ *nyelvosztály azokból az L nyelvekből áll, amelyekre létezik a fenti forgatókönyv szerinti V algoritmus, hogy*

- *ha $x \in L$, akkor $\exists \pi \in \{0,1\}^*$, amelyre $\Pr[V^\pi(x) = 1] = 1$;*
- *ha $x \notin L$, akkor $\forall \pi \in \{0,1\}^*$ esetén $\Pr[V^\pi(x) = 1] \leq 1/2$.*

A következő években több olyan eredmény született, amelyek azt látták be, hogy az 1.1. definícióban szereplő $r(n)$ és $q(n)$ függvények megfelelő rögzítésével teljesül, hogy

$\mathbf{NP} \subseteq \mathbf{PCP}[r(n), q(n)]$ [FGL$^+$91, AS92]. A végső ilyen eredmény 1992-ből származik, és PCP-tételként szokás hivatkozni rá [ALM$^+$92].

1.1. Tétel. $\mathbf{NP} \subseteq \mathbf{PCP}[\log n, 1]$

Tehát egy **NP**-beli nyelvbe tartozás nagy valószínűséggel történő eldöntéséhez elegendő $O(\log n)$ random bitet használva konstans sok helyen belenézni a bizonyításba. Sőt, Håstad [Hå01] cikke alapján elég konkrétan három bitet megnézni.

Az 1.1. tétel bizonyítása hosszú, bonyolult, és nem is tartozik minden része szorosan a jelenlegi fő témához. Annyit viszont megemlítünk róla, hogy a lényege az, hogy több lépésben konstruálunk egy egyszerű bizonyításból egy olyan, másik bizonyítást, ami már megfelel a feltételeknek. Eközben több, a dolgozatban tárgyalt témakör is fontos szerepet kap, például a linearitás és az alacsonyfokúság tesztelése, valamint az interaktív bizonyítások. Ez is mutatja, hogy szoros a kapcsolat a PCP-elmélet és a tulajdonságtesztelés között.

A tulajdonságtesztelés története a '90-es években indult; ennek áttekintéséhez Goldreich tulajdonságtesztelésről írott könyvének [Gol17] 1.4. alfejezetét használtuk. Először Blum, Ruby és Rubinfeld 1993-mas cikkében [BLR93] jelent meg a téma, ekkor még implicite, és leginkább a függvények linearitásának tesztelését mutatták be. Ezt Rubinfeld és Sudan 1996-os cikke [RS96] követte, amelyben már egy absztraktabb megközelítést használtak, az ún. robusztus karakterizációt, ami speciális esete a tulajdonságtesztelésnek.

A tulajdonságtesztelést általánosan és rendszerezetten először Goldreich, Goldwasser és Ron vizsgálta 1996-ban [GGR98]. Ők már önmagában, egy újfajta számítási problémaként tekintettek rá. Főleg gráftulajdonságok tesztelésével foglalkoztak, és náluk már előtérbe került az objektum reprezentációjának jelentősége is – a már említett cikkben a gráfok megadása szomszédsági mátrix alapon történt, míg Goldreich és Ron későbbi cikkében [GR04] szomszédsági lista alapon.

A PCP-elmélet és a tulajdonságtesztelés kapcsolata abban is tetten érhető, hogy minden PCP-konstrukció esetén szükséges egy bizonyos kódba tartozás tesztelése (a bemenet kódszó-e). Az ilyen kódokat nevezzük lokálisan tesztelhető kódoknak (locally testable codes); és ezek szoros kapcsolatban állnak a lokálisan tesztelhető bizonyítékokkal (locally testable proofs), amelyeket PCP-knek is neveznek.

1.2. Különböző PCP formalizmusok

Az 1.1. definícióban egy randomizált Turing-géppel és általa kapott bizonyítással definiáltuk a PCP-t. Ugyanakkor a történelmi eredetéhez közelebb álló módon is megfogalmazható, több-bizonyítós interaktív bizonyító rendszerként (**MIP**). Ez tekinthető egy kooperatív játéknak, amit k db játékossal játszik egy bíró. A bíró (ellenőr) kérdéseket tesz fel az egyes játékosoknak (bizonyítóknak), akik erre válaszolnak úgy, hogy közben nem kommunikálnak egymással. A játékosok célja úgy válaszolni, hogy a bíró elfogadja a válaszok összességét. Az, hogy mit fogad el a bíró, mindenki számára ismert, és a kérdésektől is függ.

Mielőtt az első kérdést elküldené a bíró, de miután az elfogadás feltétele már ismert, a játékosok megállapodhatnak egy stratégiában. Ez után minden $i \in [k]$ játékos (ahol $[k]$ az $\{1, \ldots, k\}$ halmazt jelöli) egy egyszerű f_i függvényként interpretálható: bemenete a kérdés, kimenete a válasz. Egy G játék *értékét* $\omega(G)$-vel jelöljük: ez alatt a játékosok sikervalószínűségének maximumát értjük, vagyis azt, hogy a legjobb stratégiát (melyik játékos milyen függvény szerint válaszoljon) választva milyen valószínűséggel (a feltett kérdéseken értelmezve) fogadja el a válaszaikat a bíró.

1.2. Definíció. *Egy k egész esetén egy k-résztvevős G játékot meghatároznak*

- *lehetséges kérdések véges halmazai: $Q_1, \ldots, Q_k$ (melyik játékosnak milyen kérdéseket tehet fel a bíró);*
- *egy π valószínűségi eloszlás a $Q_1 \times \cdots \times Q_k$ halmazon (melyik kérdéseket milyen valószínűséggel teszi fel a bíró);*
- *lehetséges válaszok véges halmazai: $A_1, \ldots, A_k$ (melyik játékos mit válaszolhat);*
- *döntési predikátum: $V : (A_1 \times \cdots \times A_k) \times (Q_1 \times \cdots \times Q_k) \to \{0, 1\}$ (mit fogad el a bíró).*

A játék értéke:

$$\omega(G) = \max_{f_i : Q_i \to A_i} \sum_{(q_1,\ldots,q_k) \in Q_1 \times \cdots \times Q_k} \pi(q_1, \ldots, q_k) V(f_1(q_1), \ldots, f_k(q_k), q_1, \ldots, q_k).$$

Ebben a formalizmusban a PCP-tétel a következőképpen írható.

1.2. Tétel. *Bármely $L \in$ **NP**-re létezik olyan polinom időben számítható leképezés, ami az x bemeneti karaktersorozathoz egy G_x k-résztvevős játékot rendel, ahol $k \in O(1)$, és*

- *ha $x \in L$, akkor $\omega(G_x) = 1$;*
- *ha $x \notin L$, akkor $\omega(G_x) \leq 1/2$.*

1.2.1. Megjegyzés. *A tétel másik megfogalmazása: van olyan G konstans résztvevős játék, amely esetén **NP**-nehéz feladat megkülönböztetni az $\omega(G) = 1$ esetet az $\omega(G) \leq 1/2$ esettől.*

Egy harmadik változatban is felírható a PCP, ami kényszerkielégítési problémákkal (CSP – Constraint Satisfaction Problems) kapcsolatos. Ebben a formalizmusban bináris változókon adott néhány kényszer (Boole-formula), amelyek egyenként w darab bináris változót tartalmaznak. Egy ilyen kényszerkielégítési problémát wCSP-vel jelölünk. Az a kérdés, hogy a változók tetszőleges behelyettesítésével legfeljebb hányadrésze elégíthető ki a kényszereknek. Vegyük észre, hogy $w = 3$ esetén a jól ismert Max-3SAT problémát kapjuk.

Ebben a formalizmusban a PCP-tétel:

1.3. Tétel. *Bármely $L \in$ **NP**-re létezik olyan polinom időben számítható leképezés, ami az x bemeneti karaktersorozathoz egy olyan CSP-t rendel, ahol*

- *ha $x \in L$, akkor van olyan behelyettesítése a változóknak, ami a kényszerek mindegyikét kielégíti;*
- *ha $x \notin L$, akkor nincs olyan behelyettesítése a változóknak, ami a kényszerek több mint felét kielégítené.*

1.3.1. Megjegyzés. *A tétel másik megfogalmazása: van olyan w egész és $\rho \in (0, 1)$, amely esetén egy wCSP-ről **NP**-nehéz feladat megkülönböztetni azt a két esetet, hogy a maximálisan kielégíthető kényszerek aránya 1, vagy legfeljebb ρ.*

Belátható, hogy a három tétel (az 1.1., az 1.2. és az 1.3. tétel) ekvivalens egymással. A bizonyítástól eltekintünk.

2. fejezet

Tulajdonságtesztelés

Tulajdonságtesztelésről (property testing) akkor beszélünk, amikor úgy vizsgáljuk egy adott objektumnak egy bizonyos globális tulajdonságát, hogy azt nem ellenőrizzük teljeskörűen, hanem csak lokálisan tekintünk bele az objektumba – ezért lokális tesztelésnek is nevezzük. Ekkor nem tudjuk biztosra venni, hogy jó eredményt kaptunk, csupán nagy valószínűséggel helyesen különböztetjük meg azt a két esetet, hogy az objektum teljesíti a vizsgált tulajdonságot, vagy „távol" van attól. Cserébe sokkal gyorsabb, tipikusan szublineáris lépésszámú algoritmust kapunk.

Ez szorosan kapcsolódik az ígéretproblémák, azon belül is főleg a hézagproblémák területéhez, ezért először ezeket mutatjuk be, ezután pedig a tulajdonságtesztelés alapjait ismertetjük a fejezetben.

2.1. Ígéretproblémák (promise problems)

A bonyolultságelméletben általában eldöntési problémákról beszélünk, azaz olyan problémákról, amelyek bemenetére vagy teljesül a vizsgált tulajdonság, vagy nem. Ez összhangban van a formális nyelvekkel fennálló kapcsolattal: a bemeneti szó vagy eleme a problémához tartozó nyelvnek, vagy nem: tehát az x szóra vagy $x \in L$, vagy $x \in \Sigma^* \setminus L$ teljesül, nincs harmadik lehetőség.

Ugyanakkor már itt is észrevehetjük, hogy mégis szóba jöhet egy harmadik lehetőség: mi van akkor, ha $x \notin \Sigma^*$, azaz a bemenet tartalmazhat a Σ ábécén kívüli karaktereket? Néhány konkrét példa esetén még hangsúlyosabb ez a lehetőség. Például legyen L az olyan gráfot leíró karaktersorozatok halmaza, amelyek három színnel színezhetők. Ekkor nyilvánvalóan az a lényeg, hogy a gráfok közül megkülönböztessük a három színnel színezhetőket a nem színezhetőktől. De mi lesz azokkal a bemenetekkel, amelyek nem gráfot írnak le?

Erre a szokásos válasz, az, hogy azok nincsenek L-ben, hiszen ahhoz, hogy valami egy három színnel színezhető gráf legyen, először is gráfnak kell lennie. És ez jól is működik ebben az esetben: lineáris időben ellenőrizhető, hogy a bemenet megfelel-e a gráfleíráshoz kapcsolt szintaktikai megkötéseknek. Tehát nem okoz komplexitásbeli gondot, ha először ellenőriznünk kell, hogy olyan-e a bemenet, amit várunk.

Ám ez nem minden esetben van így. A következő példában legyen L az olyan Hamilton-kört tartalmazó gráfok halmaza, amelyekben van teljes párosítás (TP). Itt már egyáltalán nem mindegy, hogy feltehetjük-e a bemenetről, hogy egy Hamilton-kört tartalmazó gráf, vagy azt nekünk kell ellenőriznünk, ugyanis a Hamilton-kör probléma **NP**-teljes. Ezért ha tudjuk, hogy a bemenet megfelel az „ígéretnek", hogy tartalmaz Hamilton-kört, akkor ahhoz, hogy eldöntsük, L-be tartozik-e, elegendő megnéznünk, hogy mi a gráf csúcsainak paritása: ha páratlan, akkor nem lehet benne TP, ha pedig páros, akkor van (a Hamilton-kör minden második éle épp TP-t alkot). Azonban ha nem bízhatunk az ígéret-

ben, akkor a TP keresése mellett (ami önmagában még eldönthető polinomiális időben) Hamilton-kört is keresnünk kell – legalábbis akkor, ha van a gráfban TP. Ugyanis az előző bekezdésben írt szokásos megoldás alapján csak TP-vel *és* Hamilton-körrel egyaránt rendelkező gráfok vannak L-ben. A Hamilton-kör keresése viszont a tudomány jelenlegi állása szerint reménytelen, hogy polinomiális időben sikerüljön.

2.1.1. Néhány definíció

Azt láttuk, hogy van olyan eset, amikor fontos különbséget jelent, hogy feltételezhetjük-e, hogy a bemenetek rendelkeznek egy adott Q tulajdonsággal. Ezt a tulajdonságot nevezzük *ígéret*nek (promise): az illető, aki megbízott minket azzal, hogy döntsük el bemenetekről, hogy rendelkeznek-e egy másik, R tulajdonsággal (avagy L-beliek-e), „megígéri" nekünk, hogy csak olyan bemeneteket kaphatunk, amelyekre a Q tulajdonság teljesül, így ennek ellenőrzésével nem kell foglalkoznunk.

Az ígéretproblémákat Even, Selman és Yacobi vezette be 1984-es cikkükben [ESY84].

2.1. Definíció. *Egy* ígéretprobléma *egy (Q, R) pár, ahol Q és R predikátumok, előbbi az ígéret, utóbbi a vizsgált tulajdonság. Egy M determinisztikus Turing-gép* megoldja *a (Q, R) problémát, ha minden x bemenetre*

$$Q(x) \Rightarrow [M(x) \text{ megáll} \wedge (M(x) = 1 \Leftrightarrow R(x))].$$

Ahol $M(x) = 1$ azt jelöli, hogy az M Turing-gép/algoritmus elfogadja az x bemenetet, avagy igenlő választ ad rá ($M(x) = 0$ pedig hasonlóan az elutasítást, a nemleges választ jelenti). Ezt a jelölést használjuk a továbbiakban is. Tehát ha teljesül az ígéret a bemenetre, akkor a Turing-gép megáll ezen a bemeneten, és pontosan akkor fogadja el, ha teljesül rá a vizsgált tulajdonság.

2.2. Definíció. *A (Q, R) ígéretprobléma* megoldható*, ha van olyan M Turing-gép, ami megoldja; ekkor az M által elfogadott $L(M)$ nyelvet a (Q, R) probléma egy* megoldás*ának nevezzük.*

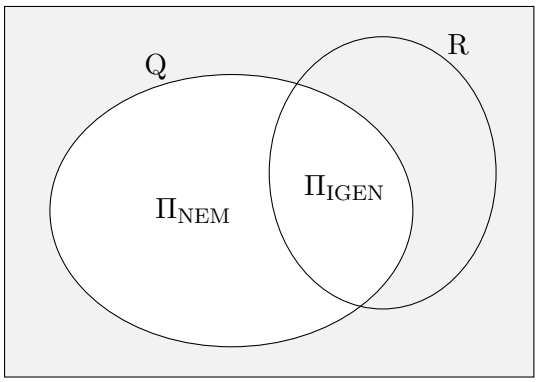
Világos, hogy egy ígéretproblémának több megoldása lehet, ugyanis a 2.1. definíció csak azokról a bemenetekről mond valamit, amelyekre teljesül a Q predikátum. Tehát az ígéretet nem teljesítő bemenetek bármelyike akár belül, akár kívül is lehet egy L megoldáson.

Az ígéretproblémák más módon is definiálhatók. Tekintsük most Q-t és R-et halmazokként, azaz azon lehetséges bemenetek halmazaiként, amelyekre teljesül az adott predikátum. Az előbbiekben Q és R segítségével definiáltuk az ígéretproblémákat. Azonban mivel úgyis csak a Q-beli elemekről mondunk valamit, tekinthetjük helyettük a $\Pi_{\text{IGEN}} = Q \cap R$ és a $\Pi_{\text{NEM}} = Q \setminus R$ halmazokat is (2.1. ábra).

Tehát a két vizsgált halmaz (avagy tulajdonság) az, hogy a bemenet, ami teljesíti az ígéretet, ezenfelül a vizsgált tulajdonságot is teljesíti-e. Ez alapján is három csoportba oszthatók a lehetséges bemenetek:

- IGEN-példányok: a Π_{IGEN}-beliek, akiket el kell fogadni, mert az ígéretnek és a vizsgált tulajdonságnak is megfelelnek;
- NEM-példányok: a Π_{NEM}-beliek, akiket nem szabad elfogadni, mert az ígéretnek megfelelnek, de a vizsgált tulajdonságnak nem;
- a nem megengedett elemek, amelyeket akár el is fogadhatunk (vagy nem), mivel nem felelnek meg ígéretnek – a 2.1. ábrán ezt a részt világosszürke háttér jelöli.

Goldreich ez alapján adott ekvivalens definíciót az ígéretproblémákra [Gol05].



2.1. ábra. Az ígéretproblémák két definíciójának kapcsolata.

2.3. Definíció. *A* Π *ígéretprobléma két diszjunkt halmazból álló pár:* $(\Pi_{\text{IGEN}}, \Pi_{\text{NEM}})$*, ahol* $\Pi_{\text{IGEN}}, \Pi_{\text{NEM}} \subseteq \Sigma^*$*, és* $\Pi_{\text{IGEN}} \cap \Pi_{\text{NEM}} = \emptyset$*. Ígéret**nek a* $\Pi_{\text{IGEN}} \cup \Pi_{\text{NEM}}$ *halmazt nevezzük.*

Látható, hogy az eredeti definícióval összhangban ígéretnek a $\Pi_{\text{IGEN}} \cup \Pi_{\text{NEM}} = Q$ halmazt nevezzük. A továbbiakban a 2.3. definíciót használjuk.

Ígéretproblémákhoz is adhatók olyan nyelvosztályok, mint az eldöntési problémák esetén például a **P**, **NP**, **BPP**[1]. Mivel a dolgozatban randomizált algoritmusokról van szó, számunkra a **BPP** nyelvosztálynak az ígéretproblémákra vonatkozó megfelelője lesz a legfontosabb, de példaként a **P** megfelelőjét is bemutatjuk.

2.4. Definíció. $\mathcal{P}$ *azon ígéretproblémák osztálya, amelyek determinisztikus algoritmussal polinomiális időben megoldhatók. Azaz* $\Pi = (\Pi_{\text{IGEN}}, \Pi_{\text{NEM}}) \in \mathcal{P}$*, ha van olyan* A *polinomidejű determinisztikus algoritmus, amire teljesülnek az alábbiak.*

- $\forall x \in \Pi_{\text{IGEN}}$ *esetén* $A(x) = 1$*;*
- $\forall x \in \Pi_{\text{NEM}}$ *esetén* $A(x) = 0$*.*

(Ismét, $A(x) = 1$ a bemenet elfogadását, $A(x) = 0$ pedig az elutasítását jelenti.) Észrevehetjük, hogy a $\Pi_{\text{IGEN}} \cup \Pi_{\text{NEM}}$ ígéreten kívüli bemenetekről ez a definíció sem mond semmit.

2.5. Definíció. $\mathcal{BPP}$ *azon ígéretproblémák osztálya, amelyek randomizált algoritmussal polinomiális időben megoldhatók. Azaz* $\Pi = (\Pi_{\text{IGEN}}, \Pi_{\text{NEM}}) \in \mathcal{BPP}$*, ha létezik olyan* A *polinomidejű randomizált algoritmus, amire teljesülnek az alábbiak.*

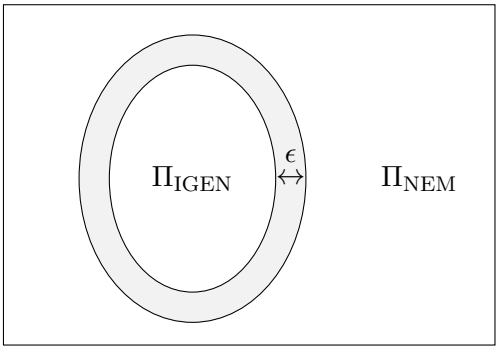
- $\forall x \in \Pi_{\text{IGEN}}$ *esetén* $\Pr[A(x) = 1] \geq 2/3$*;*
- $\forall x \in \Pi_{\text{NEM}}$ *esetén* $\Pr[A(x) = 1] \leq 1/3$*.*

Felhívjuk rá a figyelmet, hogy szándékosan jelöljük eltérő írásmóddal az eldöntési problémák és az ígéretproblémák nyelvosztályait: előbbieket félkövér, utóbbiakat kalligrafikus betűkkel írjuk. Ugyanis nem egyezik meg a két-két osztály, hanem az ígéret-változat bizonyos értelemben tartalmazza az eldöntési változatot. Hiszen minden L eldöntési problémához rendelhető egy triviális $(L, \Sigma^* \setminus L)$ ígéretprobléma, ahol az ígéret nem mond semmit, mivel $\Pi_{\text{IGEN}} \cup \Pi_{\text{NEM}} = L \cup (\Sigma^* \setminus L) = \Sigma^*$. Ekkor az L-et megoldó algoritmus $(L, \Sigma^* \setminus L)$-et is megoldja, így továbbra is teljesül L időkorlátja. Azonban a másik irányba ez nem mindig működik, erre volt példa a Hamilton-kört tartalmazó gráfok közül a teljes párosítást tartalmazók halmaza. Tehát, némileg helytelen jelöléssel, pl. $\mathbf{P} \subset \mathcal{P}$ a kapcsolat.

[1] **BPP** azon problémák osztálya, amelyek polinom időben megoldhatók randomizált algoritmussal. Avagy azon nyelveké, amelyekre létezik őket felismerő polinom időkorlátos randomizált Turing-gép.

2.1.2. Hézagproblémák (gap problems)

A hézagproblémák olyan speciális ígéretproblémák, ahol az IGEN- és a NEM-példányok között van egy „hézag”. Ez alatt azt értjük, hogy van egy, a lehetséges bemenetek halmazán értelmezett távolságmértékünk; és azt mondjuk, hogy a vizsgált tulajdonságot teljesítő elemeket (IGEN-példányok) elfogadjuk, míg az ezektől egy bizonyos ϵ távolságnál távolabb lévőket (NEM-példányok) elutasítjuk. A kettő közötti, tehát a tulajdonsággal nem rendelkező, de azokhoz elég közeli (legfeljebb ϵ távolságra lévő) elemekről bárhogy dönthetünk – a 2.2. ábrán ezek a világosszürke sávba esnek: „szürke zóna”. Tehát itt az ígéret az, hogy vagy a tulajdonsággal rendelkező elemeket kapunk, vagy olyanokat, amelyek távol vannak a tulajdonság teljesítésétől.



2.2. ábra. A hézagproblémák ábrázolása.

Az ϵ távolságparamétert általában bemenetként megkapja az algoritmus az x mellett, de adott esetben lehet egy előre rögzített globális paraméter is. Ezért az igenlő válasz többet mond, ugyanis ez nem függ a távolságparamétertől; míg a nemleges válasz mindig csak az adott ϵ-ra vonatkozik.

Ha két objektum távolsága nagyobb, mint ϵ, akkor azt mondjuk, hogy *ϵ-távol* vannak egymástól; ellenkező esetben *ϵ-közel* vannak egymáshoz. Hasonlóan definiálhatjuk egy objektum és egy tulajdonság ϵ-távolságát. Egy x objektum ϵ-távol van a T tulajdonságtól, ha bármely y-tól, ami teljesíti T-t, ϵ-távol van; ellenkező esetben x ϵ-közel van T-hez.

Tulajdonságtesztelés esetén a bemenet egy (x, ϵ) pár, ahol x a vizsgált objektum leírása, ϵ pedig a távolságparaméter. Az (x, ϵ) pár IGEN-példány, ha x rendelkezik a vizsgált tulajdonsággal; és NEM-példány, ha x ϵ-távol van a tulajdonságtól.

2.2. Tulajdonságtesztelés

Az alfejezethez felhasználtuk Goldreich gráftulajdonság-tesztelésről írott összefoglalóját [Gol10] és a tulajdonságtesztelésről írott könyvének [Gol17] 1. fejezetét.

A tulajdonságtesztelés a megfelelő eldöntési probléma relaxált változatának tekinthető. A cél, hogy a (jellemzően szublineáris futásidejű) algoritmus nagy valószínűséggel fogadja el a tulajdonsággal rendelkező objektumokat (IGEN-példányok), és szintén nagy valószínűséggel utasítsa el azokat, amelyek távol vannak a tulajdonság teljesítésétől (NEM-példányok).

2.2.1. Alapvető definíciók

Azt már említettük, hogy a tesztelő bemenete egy (x, ϵ) pár, ahol x a vizsgált objektum, ϵ pedig a távolságparaméter. De milyen módon adott az algoritmus számára x, ami lehet egy hatalmas gráf is? Ez a konkrét objektumtól függhet, de általában egy orákulumon

keresztül érhető el. Az *orákulum-hozzáférés* (oracle access) azt jelenti, hogy az objektumot egy függvényként reprezentáljuk, és az algoritmus ehhez a függvényhez intézhet lekérdezéseket: feltesz neki egy kérdést, és megkapja rá a választ, hogy ott milyen értéket vesz fel. (Vegyük észre, hogy ez megfelel a standard RAM-modellnek.) Például egy gráf esetén a tesztelő ismeri a gráf csúcsainak számát, az orákulum pedig egy csúcspárból álló kérdésre megmondja, hogy szomszédosak-e a gráfban.

Tulajdonságteszteléshez tehát két fontos dolgot kell előre meghatározni. Egyik az objektumhoz való *hozzáférés módja*, tehát az, hogy az orákulumhoz milyen kérdéseket lehet intézni. A másik egy *távolságmérték*, hogy meg tudjuk határozni a lehetséges bemeneti objektumok távolságát. Ez ahhoz szükséges, hogy megállapíthassuk, melyek azok az objektumok, amelyeket el kell utasítani, azaz ϵ-távol vannak a tulajdonságtól. A hozzáférés módja alkalmasint meghatározza a távolságmértéket.

Ezek alapján már kimondhatjuk a tulajdonságtesztelés definícióját, amelyből az is kiderül, pontosan mit értünk a korábbiakban sokszor használt „nagy valószínűséggel" kifejezés alatt.

2.6. Definíció. *Tegyük fel, hogy az A algoritmusnak orákulum-hozzáférése van az x bemeneti objektumhoz. Azt mondjuk, hogy A a T tulajdonságot tesztelő algoritmus, ha minden $\epsilon > 0$ távolságparaméter esetén teljesíti az alábbi két feltételt.*

- *Ha $x \in T$, akkor* $\Pr[A(x) = 1] \geq 2/3$.
- *Ha x ϵ-távol van T-től, akkor* $\Pr[A(x) = 1] \leq 1/3$.

A fentiekben 1/3 hibavalószínűség szerepel, de ehelyett tetszőleges 1/2-nél kisebb (de 0-nál nagyobb) konstans is megfelel, ugyanis egy tesztelő algoritmust t-szer futtatva, és a futások eredményeiből a többségi eredményt választva, t-ben exponenciálisan csökken a hibázás valószínűsége. Bizonyos tulajdonságok esetén adható olyan algoritmus is, ami az $x \in T$ bemeneteket 1 valószínűséggel elfogadja. Ez esetben azt mondjuk, hogy A-nak *egyoldali hibá*ja van. Ekkor, ha az algoritmus elutasító választ adott, akkor biztosan tudhatjuk, hogy a bemenet nem rendelkezett a T tulajdonsággal.

Az orákulum-hozzáférésből következik, hogy a tesztelő algoritmusok bonyolultságát nemcsak az idő- (és esetleg tár-) komplexitással jellemezhetjük, hanem a *lekérdezésbonyolultság*gal (query complexity) is. Egy tesztelő lekérdezéskomplexitása azt adja meg, hogy az algoritmus a futása során hányszor fordul az orákulumhoz az ϵ távolságparamétertől és esetleg az objektum méretétől függően. Egy n méretű objektum esetén n kérdés nyilván elég: a cél az, hogy n-ben szublineáris legyen a lekérdezésbonyolultság, de még jobb, ha független n-től. Ezenfelül az ϵ-tól való függését is figyeljük a lekérdezések számának, például, hogy polinomiális-e $1/\epsilon$-ban. Leggyakrabban a lekérdezés- és az időbonyolultság használatos a tulajdonságtesztelő algoritmusok komplexitásának jellemzésére. Egy tesztelő algoritmust általában *hatékony*nak neveznek, ha az időbonyolultsága polinomiálisan függ a lekérdezésbonyolultságától. Mi viszont főleg a lekérdezéskomplexitást figyeljük, így adott esetben például inkább az $1/\epsilon$-ban polinomiális kérdésszámot tekintjük hatékonynak.

További fontos jellemzője a tesztelő algoritmusnak, hogy a következő lépéseihez felhasználhatja-e a korábbi lépésekből szerzett információt. Ha igen, akkor *adaptív*, egyébként *nem adaptív* a tesztelő. Utóbbi esetben tehát csak előre meghatározott, fix lépéseket végezhet; illetve randomizált algoritmus esetén a generált véletlentől is függhet a következő lépése.

Vannak olyan tulajdonságok, amelyek tesztelhetők egy olyan alap algoritmus többszöri végrehajtásával, aminek a végrehajtása nem igényli a távolságparaméter ismeretét – csak az ismétlések száma függ tőle. Az ilyen algoritmusokat *távolságfüggetlen tesztelő*knek nevezzük (POT – Proximity Oblivious Tester). Egy T tulajdonságot tesztelő POT a T-beli elemeket legalább τ valószínűséggel elfogadja ($\tau \in (0, 1]$); de leggyakrabban egyoldali

hibája van, azaz $\tau = 1$. A nem T-beli elemek elfogadásának valószínűsége az elem T-től vett távolságával monoton csökken τ alá[2]. Egy POT lekérdezésbonyolultsága a vizsgált objektum n méretétől függhet, de néha ettől is független, azaz konstans. Belátható, hogy ha egy tulajdonságra van POT, akkor ennek a többszöri végrehajtásával standard tesztelő algoritmust kaphatunk[3]. Az így kapott tesztelőnél sok esetben adható jobb tesztelő algoritmus, illetve bizonyos tulajdonságokra van hatékony tesztelő, de nem adható jó POT.

2.2.2. A távolságparaméter szerepe

Felmerülhet a kérdés, hogy miért van szükség arra, hogy hézagproblémák tesztelését nézzük: nem lehetne-e a T-beliséről/nem T-beliségről hatékonyan, nagy valószínűséggel helyesen dönteni ahelyett, hogy a T-beli elemeket a T-től ϵ-távol lévőktől különböztetjük meg. A válasz az, hogy nem, és ez könnyen látható egy egyszerű példán. Vegyük azt a problémát, amikor egy adott, n hosszú bitsorozatban található egyesek számának a paritását szeretnénk meghatározni (azaz a vizsgált T tulajdonság például az, hogy az egyesek száma páros). A teljesen korrekt válaszhoz mind az n bitet meg kell vizsgálni, hiszen bármelyik bit az ellenkezőjére tudja fordítani a választ. Ha akár egyetlen bitet is figyelmen kívül hagyunk, 1/2-1/2 a valószínűsége annak, hogy az egyesek száma páros, illetve páratlan. Tehát ha 1/2-nél nagyobb valószínűséggel szeretnénk jól dönteni, akkor mindenképp végig kell nézni mind az n bitet. Ellenben ha a hézagprobléma változatot nézzük az $\epsilon \geq 1/n$ esetben, akkor triviális a válasz, hiszen minden n hosszú bitsorozatban egyetlen bit értékével biztosítható, hogy megfelelő legyen a paritás. Azaz bármely n hosszú bitsorozat $1/n$-közel van ahhoz, hogy páros (vagy páratlan) legyen benne az egyesek száma. Így egyetlen bitet sem kell elolvasni a bitsorozatból, egyszerűen bármely bemenetet elfogadhatja a tesztelő algoritmus. (Ha pedig $\epsilon < 1/n$, akkor az összes elem vizsgálata kevesebb mint $1/\epsilon$ kérdést jelent, ami hatékony: n-től független, $1/\epsilon$-ban pedig lineáris.)

Nézzük meg, miért elégedhetünk meg sok esetben azzal, hogy a tulajdonságot nem teljesítő, de ahhoz ϵ-közeli objektumokat minden további nélkül elfogadhatja az algoritmus.

- Gyakran elég egy közelítő megoldást adni, a közelítő algoritmusoknak is ezért van létjogosultsága.

- Előfordulhat, hogy bár teljesen megfelelő objektumra van szükség, az algoritmus futtatása után lehetőségünk van némi költség fejében kis mértékben módosítani az objektumon. Például páros gráfokat keresünk. Ha egy gráfról, amit elfogadott a tesztelő, kiderül, hogy nem páros (de nagy valószínűséggel ϵ-közel van hozzá), akkor elég néhány élet kitörölni belőle, hogy páros gráfot kapjunk. Idetartozik az az eset is, amikor zajos csatornán kapjuk az adatokat: a feladónál még páros volt a gráf, de a címzetthez már kicsit módosulva érkezik meg. Ha a kitörölt élek számával arányos a módosítás (hibajavítás) költsége, akkor az ϵ-közeliség miatt nagy valószínűséggel biztosított, hogy olcsó lesz a javítás.

- Használhatjuk a tesztelő algoritmust egy lassabb, pontos algoritmus futtatása előtt előzetes ellenőrzésre. Ha a tesztelő elutasít, akkor nem futtatjuk a lassú algoritmust – egyoldali hiba esetén ekkor teljesen biztosan elutasítaná a pontos algoritmus is. Ezzel jelentős mennyiségű időt takaríthatunk meg, különösen akkor, ha olyan eset áll fenn, ahol a bemenetek tipikusan vagy T-beliek, vagy távol vannak tőle. Sőt, ha

[2]Formálisan: ha x a vizsgált elem, aminek a T-től vett távolsága $\delta_T(x) > 0$, akkor annak a valószínűsége, hogy a POT elfogadja x-et, felülről becsülhető $\tau - \varrho(\delta_T(x))$-szel, ahol $\varrho : (0, 1] \to (0, 1]$ egy monoton növő függvény.

[3]A végrehajtások száma egyoldali hibájú POT esetén $O(1/\varrho(\epsilon))$, általános esetben $O(1/\varrho(\epsilon)^2)$, ahol ϵ az ismételt futtatásokkal kapott tesztelő távolságparamétere, ϱ pedig a fent említett monoton növő függvény.

biztosan csak ez a két eset áll fenn, akkor a tesztelő algoritmus (nagy valószínűséggel) teljesen jó eredményt ad, így lehet, hogy nem is kell futtatni a pontos algoritmust.

Megjegyezzük, hogy a tesztelésnek adható egy olyan általánosítása, amelyet *toleráns tesztelés*nek neveznek. Ebben az esetben az elfogadandó bemenetek nem csak a T-beliek, hanem a T-hez ϵ'-közeliek, ahol $\epsilon' < \epsilon$ második távolságparaméter. Tehát „toleráljuk" a T-től való nagyon kicsi eltérést. Egy ilyen tesztelőt ϵ'-*toleráns*nak nevezünk. Gyakran ϵ'-t az ϵ-nak – és időnként a vizsgált objektum n méretének – egy rögzített függvényeként adják meg (pl. $\epsilon' = \epsilon/2$). A szokásos – eddig és a továbbiakban is tárgyalt – tulajdonságtesztelés esetén $\epsilon' = 0$, azaz ezek 0-toleráns tesztelőnek tekinthetők. Sok esetben lehet jó toleráns tesztelőt adni, ugyanakkor van olyan tulajdonság, amire ugyan adható olyan standard tesztelő, amely lekérdezésbonyolultsága független az objektum n méretétől (csak ϵ-tól függ), de toleráns tesztelő esetén szükséges $n^{\Omega(1)}$ lekérdezés. A toleráns tesztelést Parnas, Ron és Rubinfeld vizsgálta elsőként [PRR06].

3. fejezet

Függvénytulajdonság-tesztelés

A függvénytulajdonságok közül a linearitást vizsgáljuk meg alaposabban, bizonyításokkal alátámasztva az eredményeket. Az alacsonyfokúságról, valamint a diktatúra és a junta tulajdonságokról már bizonyítások nélkül számolunk be. A fejezet a témakör ismert eredményeit dolgozza fel és mutatja be.

3.1. Linearitás teszt

Az itt bemutatandó eredményeket Blum, Luby és Rubinfeld írta le 1993-ban a [BLR93] cikkben. Az alfejezethez felhasznált másik forrás [RR14].

3.1. Definíció. *Legyen G és H két Abel-csoport az összeadásra. Az $f : G \to H$ függvény lineáris (pontosabban csoport homomorfizmus), ha $\forall x, y \in G$ esetén $f(x+y) = f(x)+f(y)$.*

A linearitás problémája általánosan, eldöntési problémaként megfogalmazva a következő: bemenet egy $f : G \to H$ függvény, ahol G és H Abel-csoportok (az összeadásra). Elfogadjuk a bemenetet, ha az lineáris. Az elméleti eredményeket most bináris függvényekre nézzük meg, azaz $f : \{0,1\}^k \to \{0,1\}$ esetben, de általánosíthatók általános Abel-csoportok esetére.

3.1. Tétel. *Az $f : \{0,1\}^k \to \{0,1\}$ függvény lineáris akkor és csak akkor, ha $\exists a \in \{0,1\}^k$, amelyre $\forall x \in \{0,1\}^k$ esetén $f(x) = a^T \cdot x$. (Az implicite megjelenő összeadások* mod 2 *értendők.)*

Bizonyítás. Az elégségesség bizonyítása: ha $\forall x : \ f(x) = a^T \cdot x$, akkor

$$f(x+y) = a^T \cdot (x+y) = a^T \cdot x + a^T \cdot y = f(x) + f(y).$$

A szükségesség bizonyítása: jelöljük a standard bázisvektorokat a következőképpen: $e_1 = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, $e_2 = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}$, ..., $e_k = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}$. Legyen az a vektor $a = \begin{pmatrix} f(e_1) \\ f(e_2) \\ \vdots \\ f(e_k) \end{pmatrix}$. Így bármely $x \in \{0,1\}^k$ esetén, amelynek az $I \subseteq \{1, 2, \ldots, k\}$ halmazbeli koordinátái egyesek, a többi nulla, $f(x) = f(\sum_{i \in I} e_i) = \sum_{i \in I} f(e_i) = a^T \cdot x$.

A második egyenlőségnél felhasználtuk, hogy $f(x+y) = f(x) + f(y)$. A harmadik egyenlőség pedig azért igaz, mert a bal oldalán az a vektornak épp azokat az elemeit adjuk össze, amelyeknek megfelelő x-beli elemek 1-esek. □

Azt láttuk tehát be, hogy az ilyen értelemben lineáris függvények $f(x) = a^T \cdot x$ alakúak (nem pedig $f(x) = a^T \cdot x + b$, ahogy általában a lineáris függvényekre gondolunk), azaz mindenképp átmennek az origón (a nullvektorhoz 0-t rendelnek).

3.1.1. A tesztelő algoritmus

Ahhoz, hogy módszert adjunk arra, hogy egy adott $f : \{0,1\}^k \to \{0,1\}$ függvényről eldöntsük, hogy lineáris-e, meg kell mondanunk, hogy mit értünk egy függvény megadása alatt. Esetünkben a következő orákulumos megadási módot támogatjuk: f egy fekete dobozként (black box) adott. Ez alatt azt értjük, hogy rendelkezésünkre áll egy eljárás, aminek a belső működését ugyan nem ismerjük, de tudjuk, hogy tetszőleges $x \in \{0,1\}^k$ értéket adva a bemenetére $f(x)$-et adja kimenetként.

Az alap algoritmus nagyon egyszerű:

Bemenet: Orákulum-hozzáférés az $f : \{0,1\}^k \to \{0,1\}$ függvényhez.

A fő lépések:

1. Sorsoljunk a $\{0,1\}^k$ halmazból egyenletes eloszlással két elemet: x és y.
2. **Ha** $f(x+y) = f(x) + f(y)$, **akkor** fogadjunk el; **egyébként** utasítsunk el.

3.1. Algoritmus: Az alap algoritmus (POT) linearitás teszthez.

Ez az algoritmus önmagában nyilván könnyedén adhat helytelen eredményt, ha f nem lineáris, ezért sokszor megismételjük. Ha bármelyik futás eredménye elutasítás, akkor nem kell többször futtatni: találtunk egy ellenpéldát, ami tanúsítja, hogy a függvény nem lineáris. De meddig futtassuk, ha nem talál ellenpéldát (vagy azért, mert valóban lineáris a függvény, vagy azért, mert „majdnem lineáris")? Ez a linearitástesztelés legfontosabb eredménye.

3.1.2. A témakör fő tétele

A fő eredmény szabadon megfogalmazva azt mondja ki, hogy ha egy függvény távol van a linearitástól, akkor elég nagy valószínűséggel elutasítja a fenti 3.1. algoritmus. A tétel pontos kimondásához definiálnunk kell egy távolságmértéket függvények között.

3.2. Definíció. *Az $f, g : \{0,1\}^k \to \{0,1\}$ függvények távolsága:* $d(f,g) = \Pr[f(x) \neq g(x)]$.

Itt a valószínűséget a bemeneteken vesszük, azaz a távolság azt adja meg, hogy az összes lehetséges bemenet hányadrészén ad más kimenetet a két függvény. Ezt felhasználva egy függvény távolsága a linearitástól, avagy a lineáris függvényektől:

3.3. Definíció. *Az $f : \{0,1\}^k \to \{0,1\}$ függvénynek a lineáris függvényektől vett távolsága:* $d(f, \mathrm{lin}) = \min_{g \in \mathrm{lin}} d(f,g)$, *ahol g a $\{0,1\}^k \to \{0,1\}$ lineáris függvények közül való.*

Ez a mérték tehát azt adja meg, hogy legkevesebb hány helyen – pontosabban a helyek hányadrészén – kellene módosítani a függvény értéket ahhoz, hogy egy lineáris függvényt kapjunk. Ezzel a fogalommal már precízen ki tudjuk mondani a tételt.

3.2. Tétel. *Bármely $\epsilon \in (0,1)$ esetén, ha $d(f, \mathrm{lin}) > \epsilon$, akkor* $\Pr[\text{elut.}] > \min\{\frac{2}{9}, \frac{\epsilon}{2}\}$.

Ahol Pr[elut.] annak a valószínűségét jelöli, hogy a 3.1. algoritmus elutasító választ ad (egyszeri futtatásakor). Vegyük észre, hogy azt állítjuk, hogy az alap algoritmus egy egyoldali hibájú POT.

A 3.2. tétel tehát azt mondja ki, hogy kis ϵ ($0 < \epsilon < \frac{4}{9}$) esetén legalább $\epsilon/2$ valószínűséggel elutasít az algoritmus, azaz várható értékben $2/\epsilon$ futtatás után kapunk elutasító

választ. (Hiszen a kísérletünk az, hogy addig futtatjuk az $\epsilon/2$ valószínűséggel elutasító algoritmust, amíg elutasítást nem kapunk. Tehát a futtatások száma $\epsilon/2$ paraméterű geometriai eloszlású valószínűségi változó, aminek a várható értéke a paraméter reciproka.)

A 3.2. tétel bizonyítása több lépésben történik. Definiálunk egy g függvényt, amiről először belátjuk, hogy nincs túl távol f-től, majd pedig azt, hogy lineáris (ez utóbbi lépés két lemmából áll). Ezekből már következni fog a tétel állítása. Legyen

$$g(x) = \mathrm{maj}_{y\in\{0,1\}^k}(f(x+y)+f(y)), \tag{3.1}$$

ahol maj a többségi értéket jelenti. Nézzük meg, mit is jelent ez a g-t meghatározó képlet. Ha f lineáris, akkor $\forall y \in \{0,1\}^k$ esetén $f(x+y)+f(y) = f(x)$, hiszen bináris függvényről és mod 2 összeadásról van szó, azaz $f(x+y)+f(y)$ helyett írható $f(x+y)-f(y)$, tehát $g(x) = f(x)$. Ha nem lineáris, akkor $g(x)$ értéke (a 0 és az 1 közül) az, amit $f(x)$ értékeként tekintve f több y esetén mutat linearitást az x-szel párosítva. Úgy is fogalmazhatunk, hogy minden lehetséges y bemenet „szavaz" 0-ra vagy 1-re attól függően, hogy ő $f(x)$ melyik értéke esetén mutat linearitást x-szel párban vizsgálva. Végül $g(x)$ értéke az lesz, amire a többség szavazott.

3.3. Lemma. $d(f,g) \le 2 \cdot \Pr[\text{elut.}]$.

Bizonyítás. A teljes valószínűség tétele szerint

$$\Pr[\text{elut.}] = \Pr[\text{elut.}|f(x)\neq g(x)]\cdot\Pr[f(x)\neq g(x)] + \Pr[\text{elut.}|f(x)=g(x)]\cdot\Pr[f(x)=g(x)].$$

Mivel a második tag nemnegatív, $\Pr[\text{elut.}] \ge \Pr[\text{elut.}|f(x)\neq g(x)]\cdot\Pr[f(x)\neq g(x)]$.

A függvények távolságának definíciója szerint $\Pr[f(x)\neq g(x)] = d(f,g)$. A g függvény (3.1) definíciója alapján $f(x)\neq g(x)$ jelentése, hogy az y lehetséges értékeinek többségére $f(x+y)\neq f(x)+f(y)$. Ez épp azt jelenti, hogy $f(x)\neq g(x)$ esetén a 3.1. algoritmus legalább $1/2$ valószínűséggel elutasít. Tehát azt kaptuk, hogy

$$\Pr[\text{elut.}] \ge \Pr[\text{elut.}|f(x)\neq g(x)]\cdot\Pr[f(x)\neq g(x)] \ge 1/2\cdot d(f,g),$$

amiből átrendezéssel következik az állítás. □

3.4. Lemma. *Tetszőleges* $\delta \in (0, \frac{1}{4}]$ *esetén ha* $\Pr[\text{elut.}] \le \delta$, *akkor* $\forall x \in \{0,1\}^k$*-ra* $\Pr[g(x) = f(x+y)-f(y)] > \frac{1+\sqrt{1-4\delta}}{2}$.

Bizonyítás. Legyen p annak a valószínűsége, hogy $g(x)$ értékének meghatározásakor egy tetszőleges $y \in \{0,1\}^k$ lehetséges bemenet a többség mellett „szavaz". Ez azt jelenti, hogy $p = \Pr[g(x) = f(x+y)-f(y)]$ és $1-p = \Pr[g(x)\neq f(x+y)-f(y)]$. Úgy is fogalmazhatunk, hogy p azt adja meg, hogy a lehetséges bemenetek hányadrésze szavaz $g(x)$-re. Ezért $p \ge 1/2$, hiszen $g(x)$ a többség szavazata.

Legyen az α annak az eseménynek a valószínűsége, hogy két tetszőleges bemenet (pl. y és z) ugyanarra az értékre szavaz, vagyis $\alpha = \Pr[f(x+y)-f(y) = f(x+z)-f(z)]$. Ez felírható az előbb bevezetett p segítségével is: két bemenet pontosan akkor szavaz ugyanarra, ha vagy mindkettő $g(x)$-re szavaz, vagy egyikük sem, tehát $\alpha = p^2+(1-p)^2$.

Az α definíciójában szereplő $f(x+y)-f(y) = f(x+z)-f(z)$ kifejezés – felhasználva, hogy bináris függvényről van szó, azaz az összeadás művelet megegyezik a kivonással – átírható $f(y)+f(z) = f(x+y)+f(x+z)$ alakba, ezért α is átírható: $\alpha = \Pr[f(y)+f(z) = f(x+y)+f(x+z)]$.

Tudjuk, hogy $\Pr[\text{elfogad}] = 1-\Pr[\text{elut.}] \ge 1-\delta$. A $\Pr[\text{elfogad}]$ annak a valószínűsége, hogy két tetszőleges bemenet linearitást mutat. Ez a két tetszőleges bemenet lehet pl. y és z vagy $x+y$ és $x+z$, így $\Pr[\text{elfogad}] = \Pr[f(y)+f(z) = f(y+z)] = \Pr[f(x+y)+f(x+z) = f(y+z)] \ge 1-\delta$.

Ezeket felhasználva kapjuk:

$$\begin{aligned}\alpha = \Pr[f(y)+f(z) = f(x+y)+f(x+z)] \geq \\ \Pr[f(y)+f(z) = f(y+z) \cap f(x+y)+f(x+z) = f(y+z)] = \\ 1-\Pr[f(y)+f(z) \neq f(y+z) \cup f(x+y)+f(x+z) \neq f(y+z)] \geq \\ 1-(\delta^2+2\delta(1-\delta)) > \\ 1-2\delta.\end{aligned}$$

Tehát $\alpha = p^2+(1-p)^2 > 1-2\delta$, ami átrendezve $p^2-p+\delta > 0$. Ennek megoldása, felhasználva, hogy $p \geq 1/2$, $p > \frac{1+\sqrt{1-4\delta}}{2}$, ami épp a belátandó állítás. □

3.5. Lemma. *Ha* $\Pr[\text{elut.}] \leq \frac{2}{9}$, *akkor* g *lineáris.*

Bizonyítás. A 3.4. lemmát $\delta = 2/9$-re alkalmazva kapjuk, hogy a $\Pr[\text{elut.}] \leq \frac{2}{9}$ feltétel teljesülése esetén – a valószínűségeket z-re nézve, azaz x és y fix, de tetszőleges – igaz az alábbi három egyenlőtlenség (hiszen $\frac{1+\sqrt{1-4\cdot 2/9}}{2} = 2/3$).

$$\begin{aligned}\Pr[g(x) = f(x+z)+f(z)] > 2/3 \\ \Pr[g(y) = f(y+z)+f(z)] > 2/3 \\ \Pr[g(x+y) = f(y+z)+f(x+z)] > 2/3\end{aligned}$$

A harmadik egyenlőtlenségben a z szerepét $x+z$ veszi át – ha z bejárja a lehetséges bemenetek halmazát, akkor a fix x-hez adva is ugyanezt teszi –, és $f(x+y+x+z) = f(y+z)$.

Ezért a fenti események komplementereinek valószínűségeire igazak a következők.

$$\begin{aligned}\Pr[g(x) \neq f(x+z)+f(z)] < 1/3 \\ \Pr[g(y) \neq f(y+z)+f(z)] < 1/3 \\ \Pr[g(x+y) \neq f(y+z)+f(x+z)] < 1/3\end{aligned}$$

Ezeket összeadva, és felhasználva, hogy $\Pr[A \cup B] \leq \Pr[A]+\Pr[B]$ (Boole-egyenlőtlenség), a következőt kapjuk.

$$\Pr[g(x) \neq f(x+z)+f(z) \cup g(y) \neq f(y+z)+f(z) \cup g(x+y) \neq f(y+z)+f(x+z)] < 1$$

Ez azt jelenti, hogy z helyébe helyettesíthető olyan z_0 érték, amire teljesül az alábbi három egyenlet.

$$\begin{aligned}g(x) = f(x+z_0)+f(z_0) \\ g(y) = f(y+z_0)+f(z_0) \\ g(x+y) = f(y+z_0)+f(x+z_0)\end{aligned}$$

Az első kettőt összeadva $g(x)+g(y) = f(x+z_0)+f(y+z_0)$. A kapott jobb oldal éppen ugyanaz, mint a harmadik egyenlet jobb oldala, ezért a bal oldalak is megegyeznek, azaz $g(x+y) = g(x)+g(y)$. Mivel x és y tetszőleges, beláttuk az állítást. □

A három lemmából következik a 3.2. tétel állítása: ha $\Pr[\text{elut.}] \leq \frac{2}{9}$, akkor a 3.5. lemma miatt g lineáris. Ezért először a 3.3. lemmát, majd g linearitását, végül a 3.2. tétel feltételét felhasználva kapjuk, hogy $\Pr[\text{elut.}] \geq \frac{d(f,g)}{2} \geq \frac{d(f,\text{lin})}{2} > \frac{\epsilon}{2}$. Tehát ha $\Pr[\text{elut.}] \leq \frac{2}{9}$, akkor $\Pr[\text{elut.}] > \frac{\epsilon}{2}$ is igaz, egyébként pedig nyilván $\Pr[\text{elut.}] > \frac{2}{9}$. Összefoglalva, azt kaptuk, hogy $\Pr[\text{elut.}] > \min\{\frac{2}{9}, \frac{\epsilon}{2}\}$, ami épp a tétel állítása.

A tételből következik, hogy a nemlinearitást mutató bemenetpárok és az összes lehetséges bemenetpár hányadosának reciprokával (azaz $1/\epsilon$-nal) arányos számú iterációt végrehajtva a 3.1. algoritmusból, nagy valószínűséggel találunk bizonyítékot a függvény nemlinearitására.

Valóban, ha k-szor hajtjuk végre az alap algoritmust (a POT-ot) egy, a linearitástól ϵ-távoli függvényen, akkor a tétel szerint annak a valószínűsége, hogy nem derül ki a nemlinearitás, $(1-\epsilon/2)^k$. A tulajdonságtesztelséhez az kell, hogy ez (legfeljebb) $1/3$ legyen, azaz $(1-\epsilon/2)^k = 1/3$, amiből

$$k = \log_{1-\epsilon/2} 1/3 = \frac{-\log_2 3}{\log_2 \frac{2-\epsilon}{2}} = \frac{\log_2 3}{1-\log_2(2-\epsilon)} \leq \frac{\log_2 3}{\frac{\epsilon}{2\ln 2}} \leq \frac{2{,}2}{\epsilon}.$$

Az utolsó előtti lépésben azt használtuk, hogy $\epsilon \in [0,1]$ esetén $\epsilon + 2^{1-\frac{\epsilon}{2\ln 2}} \geq 2$ teljesül[1], amiből $1-\log_2(2-\epsilon) \geq \frac{\epsilon}{2\ln 2}$ következik.

Mivel a POT lekérdezéskomplexitása konstans (konkrétan 3), és $O(1/\epsilon)$ futtatás elég ahhoz, hogy tulajdonságtesztelőt kapjunk, a tesztelő lekérdezésbonyolultsága $O(1/\epsilon)$.

A linearitás teszt algoritmusát megvalósítottunk és a futtatások során szerzett tapasztalatokat értelmeztük. Ezek az eredmények a 7.1. szakaszban olvashatók.

3.2. Alacsonyfokúság tesztelése

Ebben a szakaszban az imént bemutatott linearitás teszt egy általánosítását vizsgáljuk, ahol az elfogadott függvények fokszáma egynél magasabb is lehet. Ez azt jelenti, hogy nemcsak lineáris függvényeket, hanem legfeljebb d fokú polinomokat is elfogadhatunk valamely d-re. A hosszú bizonyításoktól ezúttal eltekintünk, csak a fő gondolatokat, összefüggéseket mutatjuk be. Ennek a témának a fő eredményeit Rubinfeldnek és Sudannak köszönhetjük [RS96], ezenkívül forrásként felhasználtuk Goldreich munkáját is [Gol17, 3. fejezet].

3.2.1. Egyváltozós eset

A linearitás teszthez hasonlóan járhatunk el. Itt is egy POT-ot adunk, amelyet többször végrehajtva kapjuk a tesztelő algoritmust. Tudjuk, hogy $d+1$ független pont egyértelműen meghatároz egy d fokú polinomot. Ezért $d+1$ pontot mintavételezve a vizsgált függvényből megnézzük, hogy melyik polinomot határozzák meg, majd még egy pontot lekérdezünk, és ha ez illeszkedik a többi polinomjára, akkor elfogadunk, egyébként elutasítunk. Az nyilvánvaló, hogy ha f valóban legföljebb d fokú, akkor ez a teszt 1 valószínűséggel elfogad.

Nézzük ezt a tesztet formálisabban. Legyen $\mathcal{F}$ egy prím méretű véges test, a vizsgált függvény pedig $f : \mathcal{F} \to \mathcal{F}$. Azt mondjuk, hogy f alacsony fokú, ha egy $d < |\mathcal{F}|/2$ fokú polinom. Az f függvény értékeit lekérdezzük $d+2$ helyen, és megnézzük, hogy van-e olyan d fokú polinom, amire mind illeszkednek. Ha a $d+2$ pontot egymástól függetlenül sorsolnánk, az $(d+2)\log_2(|\mathcal{F}|)$ random bit generálását igényelné. Ehelyett elegendő két $r, s \in \mathcal{F}$ számot sorsolni az egyenletes eloszlásból, és ezekből előállítani a $d+2$ vizsgált helyet: $\{r, r+s, \ldots, r+(d+1)s\}$. (Például a PCP-k esetén fontos lehet a felhasznált véletlen bitek számát korlátozni.) Erről a változatról azonban jóval bonyolultabb megindokolni, hogy miért utasítja el nagy valószínűséggel a távoli függvényeket.

De hogyan tudjuk megmondani, hogy a $d+2$ pont egy d fokú polinomra esik-e? Ebben segít az alábbi tétel.

3.6. Tétel. *Az $f : \mathcal{F} \to \mathcal{F}$ függvény egy legfeljebb d fokú polinom akkor és csak akkor, ha minden $r, s \in \mathcal{F}$ esetén $\sum_{i=0}^{d+1} \alpha_i f(r+si) = 0$, ahol $\alpha_i = (-1)^{i+1}\binom{d+1}{i}$.*

[1] Ez onnan látszik, hogy $\epsilon = 0$-ra teljesül az állítás, és ha $\epsilon > 0$, akkor a bal oldal függvénye monoton nő, hiszen ekkor a deriváltja pozitív: $1 - e^{-\epsilon/2}$.

Ez a tétel lényegében arról szól, hogy egy d fokú polinom értéke egy pontban meghatározható $d+1$ másik pontbeli értékből. Hiszen például $i=0$ és $\alpha_0=-1$ helyettesítéssel azt kapjuk, hogy $f(r)=\sum_{i=1}^{d+1}\alpha_i f(r+si)$.

Van azonban egy sokkal egyszerűbb indoklás is, ami cserébe elvárja, hogy az utolsó, $(d+2)$-edik szám (amiről ellenőrizzük, hogy a többi által meghatározott polinomon van-e) a többi $d+1$ szám kizárásával, az egyenletes eloszlásból véletlenszerűen választassék. Ekkor a helyes működés indoklása a következő. Ha f ϵ-távol van a legfeljebb-d-fokúságtól, az definíció szerint azt jelenti, hogy minden legföljebb d fokú polinomhoz viszonyítva, a lehetséges bemenetek több mint ϵ részében más értéket vesz fel. Mivel az első $d+1$ pont által meghatározott f' függvény egy ilyen polinom, következik, hogy f és f' távolsága nagyobb, mint ϵ. Tehát a $(d+2)$-edik pontot egyenletesen véletlenszerűen választva, ϵ-nál nagyobb eséllyel derül ki, hogy $f\neq f'$, vagyis ϵ-nál nagyobb valószínűséggel az algoritmus elutasít.

3.2.2. Többváltozós eset

Az első fontos kérdés, ami felmerül, hogy mit értünk egy többváltozós polinom fokszáma alatt. Egy gyakori definíció, amit mi is használunk, azt mondja, hogy egy polinom foka a monomjai fokainak maximuma, egy monom foka pedig a benne szereplő változók kitevőinek összege. Például az $a^4b^3+a^3c^3$ polinom foka $\max\{4+3;3+3\}=7$. Ezt teljes foknak (total degree) is nevezik, megkülönböztetendő a változónként külön számított fokszámoktól.

Egy m változós $f:\mathcal{F}^m\to\mathcal{F}$ függvényt vizsgálunk. A legfontosabb eredmény a témában azt mondja ki, hogy egy (többváltozós) polinom pontosan akkor legfeljebb d fokú, ha bármely egyenesre megszorítva, legfeljebb d fokú (egyváltozós) polinomot kapunk. Ez az állítás a következőképpen formalizálható.

3.4. Definíció. *Egy $\mathcal{F}^m$-beli $l_{x,h}$* egyenes *alatt, ahol $x,h\in\mathcal{F}^m$, az $\{x+hi : i\in\mathcal{F}\}$ ponthalmazt értjük. Azt mondjuk, hogy ez az egyenes h eltolással átmegy x-en.*

3.7. Tétel. *Egy $f:\mathcal{F}^m\to\mathcal{F}$ függvény pontosan akkor legfeljebb d fokú polinom, ha minden $x,h\in\mathcal{F}^m$ esetén a $g_{x,h}(i)=f(x+hi)$ (ahol $i\in\mathcal{F}$) egyváltozós függvény – ami f megszorítása az $l_{x,h}$ egyenesre – egy legfeljebb d fokú polinom.*

A 3.7. tételből már sejthető, mi lesz az algoritmus (a POT): egyenletesen véletlenszerűen választunk $x,h\in\mathcal{F}^m$ értékeket, majd a 3.6. tétel segítségével ellenőrizzük, hogy f az $l_{x,h}$ egyenesre megszorítva egy legfeljebb d fokú polinom-e. A két tételt összefoglalva úgy is fogalmazhatunk, hogy azt ellenőrizzük, hogy $\sum_{i=0}^{d+1}\alpha_i f(x+hi)=0$ teljesül-e, ahol $\alpha_i=(-1)^{i+1}\binom{d+1}{i}$.

Bemenet:
Orákulum-hozzáférés az $f:\mathcal{F}^m\to\mathcal{F}$ függvényhez;
$d<|\mathcal{F}|/2$ fokszám.

A fő lépések:

1. Sorsoljunk az $\mathcal{F}^m$ halmazból egyenletes eloszlással két elemet: x és h.
2. Legyen minden $i\in\{0,1,\ldots,d+1\}$ esetén $\alpha_i=(-1)^{i+1}\binom{d+1}{i}$.
3. **Ha** $\sum_{i=0}^{d+1}\alpha_i f(x+hi)=0$, **akkor** fogadjunk el; **egyébként** utasítsunk el.

3.2. Algoritmus: Az alap algoritmus (POT) alacsonyfokúság teszthez.

Belátható, hogy a 3.2. algoritmus a legfeljebb-d-fokúságtól ϵ-távoli függvényeket legalább $\min\{\epsilon,1/(d+2)^2\}/2$ valószínűséggel elutasítja. Tehát valóban POT-ról beszélünk,

aminek többszöri végrehajtásával tesztelő algoritmust kapunk a „legfeljebb d fokú polinomság” tulajdonságra.

3.3. Diktatúra és junta

A diktatúra és a junta jól ismert fogalmak a politika árnyékos oldaláról. Itt azonban más, de az eredeti jelentéshez kapcsolódó értelemben használjuk őket függvények jellemzésére. Az alfejezet megírásához elsősorban Blais írásait használtuk fel [Bla09, Bla10].

3.5. Definíció.

- *Egy m változós $f : \{0,1\}^m \to \{0,1\}$ Boole-függvényt diktatúrának nevezünk, ha egyetlen változója meghatározza az értékét, azaz ha létezik olyan $i \in [m]$ és $b \in \{0,1\}$, amelyekre $\forall x \in \{0,1\}^m : \ f(x) = x_i \oplus b$ (ahol x_i az x vektor i-edik bitjét, $\oplus$ pedig a kizáró vagy műveletet jelöli).*
- *Egy m változós $f : \{0,1\}^m \to \{0,1\}$ Boole-függvényt k-juntának nevezünk, ha k változója meghatározza az értékét, azaz ha létezik olyan $S \subseteq [m]$, $|S| = k$ és olyan $f' : \{0,1\}^k \to \{0,1\}$, amire $\forall x \in \{0,1\}^m : \ f(x) = f'(x_S)$ (ahol $x_S \in \{0,1\}^k$ az x vektornak pontosan az S-beli indexekhez tartozó bitjeit tartalmazza, mégpedig index szerint növekvő sorrendben).*

Látható, hogy a diktatúra éppen megfelel az 1-juntának, ezért ha általános k-juntára van tesztelő algoritmus, az a $k = 1$ esetben éppen diktatúratesztelő.

Először Bellare, Goldreich és Sudan kutatták a diktatúra függvények tesztelését [BGS95]. Ők vezették be a PCP-konstrukciókban fontos szerepet játszó Hosszú Kódot (Long Code), ami egy diktatúra függvénynek is tekinthető[2].

A k-junta-tesztelés témájával először Fischer és társai foglalkoztak [FKR$^+$04]. Adtak egy $\tilde{O}(k^2/\epsilon)$ lekérdezésbonyolultságú tesztelő algoritmust (ahol $\tilde{O}(f(n))$ valamely c-re az $O(f(n)\log^c f(n))$ rövidített jelölése), és azt is belátták, hogy legalább $\Omega(\sqrt{k})$ kérdésre szükség van. Később mindkét oldalról történt javítás az eredményeken: a jelenlegi legjobb korlátok szerint kell legalább $\Omega(k)$ kérdés [CG04], és $O(k \log k + k/\epsilon)$ már elég [Bla09]. Ez utóbbi eredményhez tartozó algoritmust mutatjuk be az alábbiakban.

Jelölje $x, y \in \{0,1\}^m$ és $S \subseteq [m]$ esetén $x_{\bar{S}}y_S \in \{0,1\}^m$ azt a vektort, amit úgy kapunk, hogy x-ben az S-beli indexekhez tartozó biteket a megfelelő y-beli bitekre cseréljük. Egy $S \subseteq [m]$ indexhalmazt *befolyásos*nak nevezünk, ha $\exists x, y \in \{0,1\}^m$, amelyekre $f(x) \neq f(x_{\bar{S}}y_S)$.

Az algoritmus alapgondolata, hogy az $[m]$ indexhalmaznak egy (elég sok halmazt tartalmazó) $\mathcal{R}$ partícióját vesszük[3], és azt próbáljuk megbecsülni, hogy a particionáló halmazok közül hány befolyásos van. Ehhez kezdetben kiindulunk az $S = [m]$ indexhalmazból: ennek a befolyásosságát vizsgáljuk egy véletlen $(x, y) \in \{0,1\}^m$ pár sorsolásával. Ha S a teszt során befolyásosnak találtatik (azaz $f(x) \neq f(x_{\bar{S}}y_S)$), akkor bináris keresés-sel keresünk egy olyan S-beli particionáló halmazt, ami szintén befolyásos. Ezután S-ből elhagyjuk ezt a particionáló halmazt, és az így kapott halmaz lesz S új értéke: ennek a befolyásosságát vizsgáljuk a következő iterációban. Ha végül több mint k befolyásos particionáló halmazt találtunk, akkor elutasítunk, egyébként elfogadunk. A 3.3. algoritmus részletesen ismerteti a működést.

[2] A Hosszú Kód definiálásához először vegyük észre, hogy egy $\ell \in [L]$ címke $\log L$ bittel leírható, azaz $\ell \in \{0,1\}^{\log L}$. Jelölje $v \in \{0,1\}^L$ és $i \in \{0,1\}^{\log L}$ esetén $v(i)$ a v vektor i-edik bitjét. Egy $\ell \in [L]$ címkéhez rendelt Hosszú Kód egy $f_\ell \in \{0,1\}^{2^L}$ vektor, amire minden $x \in \{0,1\}^L$ esetén teljesül, hogy $f_\ell(x) = x(\ell)$. Ebből látszik, hogy a Hosszú Kód egy $f_\ell : \{0,1\}^L \to \{0,1\}$ függvényként diktatúra: a bemenetének ℓ-edik bitje önmagában meghatározza a kimenetet.

[3] Egy A halmaznak $\{X_1, \ldots, X_n\}$ partíciója, ha $\cup_{i\in[n]} X_i = A$ és $\forall i \neq j : X_i \cap X_j = \emptyset$ teljesül.

Bemenet:
Orákulum-hozzáférés az $f : \{0,1\}^m \to \{0,1\}$ függvényhez;
k paraméter;
ϵ távolságparaméter.

A fő lépések:

1. Particionáljuk $[m]$-et véletlenszerűen $|\mathcal{R}| = 10^{20}k^9/\epsilon^5$ részre: $(R_1, \ldots, R_{|\mathcal{R}|})$.
2. Legyen $S = [m]$ és $\ell = 0$.
3. **Ciklus** $12(k+1)/\epsilon$ körön át
   - 3.1. Sorsoljunk véletlenszerűen egy $(x, y) \in \{0,1\}^m \times \{0,1\}^m$ párt.
   - 3.2. **Ha** $f(x) \neq f(x_{\bar{S}}y_S)$, **akkor**
     - 3.2.1. Keressünk egy befolyásos $R_j \subseteq S$ halmazt bináris kereséssel.
     - 3.2.2. $S \leftarrow S \setminus R_j$ és $\ell \leftarrow \ell + 1$.
     - 3.2.3. **Ha** $\ell > k$, **akkor** utasítsunk el.
4. Fogadjunk el.

3.3. Algoritmus: k-junta-tesztelő algoritmus.

A 3.2.1. lépésben a bináris keresés úgy működik, hogy az S-beli particionáló halmazok egyik felének az unióját, S'-t teszteljük a befolyásosság szempontjából ugyanazzal az (x, y) párral, amivel S-t vizsgáltuk. Ha befolyásosnak mutatkozik (azaz $f(x) \neq f(x_{\bar{S}'}y_{S'})$), akkor S' valamely részhalmazának megfelelő indexeken is elegendő x-et y-ra módosítani ahhoz, hogy a függvény értéke „átbillenjen”, így S' partícióját felezzük tovább. Ellenkező esetben még az S'-n felül az $S \setminus S'$ valamely részét is y szerintire kell változtatni az érték átbillentéséhez, így $S \setminus S'$-t felezzük tovább úgy, hogy közben az S'-beli koordináták y szerintiek maradnak. Végül egy olyan $\mathcal{R}$-beli particionáló halmazt találunk $\log(|\mathcal{R}|)$ lépésben, aminek x szerintiről y szerintire változtatása átbillenti f értékét, azaz befolyásos.

A 3.2. lépésben összesen legfeljebb $2 \cdot 12(k+1)/\epsilon \in O(k/\epsilon)$ lekérdezés történik, a 3.2.1. lépésben pedig $(k+1)\log(|\mathcal{R}|) \in O(k\log(k/\epsilon))$, ami összesen $O(k\log k + k/\epsilon)$. Belátható, hogy ez az algoritmus valóban k-junta-tesztelő. Az világos, hogy minden k-juntát mindenképp elfogad, tehát egyoldali hibája van. A másik irányt bonyolultabb igazolni, azt a 3.8. tétel mondja ki.

3.8. Tétel ([Bla09] Lemma 3.1 és Theorem 1.1)**.** *Legyen $\mathcal{R} = (R_1, \ldots, R_t)$ egy véletlenszerű partíciója $[m]$-nek $10^{20}k^9/\epsilon^5$ részre, amelyet úgy kapunk, hogy egymástól függetlenül, egyenletesen rendelünk minden elemet egy részhez. Ha az $f : \{0,1\}^m \to \{0,1\}$ függvény ϵ-távol van attól, hogy k-junta legyen, akkor legfeljebb 1/3 valószínűséggel fogadja el a 3.3. algoritmus.*

4. fejezet

Gráftulajdonság-tesztelés

A gráfok rendkívül hasznosak, amikor olyan objektumot modellezünk, amelyben egymással bináris kapcsoltban álló entitások vannak. A mai világban, ahol a „big data" jelentősége egyre nagyobb, számos olyan kiterjedt hálózati struktúrával találkozunk, amelyek rengeteg adatot hordoznak magukban. Ezeknek a hálózatoknak fontos ismerni bizonyos globális tulajdonságait, de ezek pontos, kimerítő módon történő meghatározása legtöbbször reménytelen.

Ezért a tulajdonságtesztelés módszert használhatjuk, azaz a gráf csak egy kis részét vizsgáljuk, lokálisan tesztelünk, és ebből következtetünk a globális tulajdonság meglétére. Ahogy azt a tulajdonságtesztelésnél már említettük (2.2. szakasz), ehhez meg kell adnunk egy hozzáférési módot (milyen függvényként reprezentáljuk a gráfot) és egy távolságmértéket. Ezek alapján három modellről teszünk említést, az egyikkel közülük pedig bővebben is foglalkozunk. A fejezet egészéhez felhasználtam Goldreich gráftulajdonságok teszteléséről szóló írását [Gol10].

A továbbiakban gráftulajdonság alatt olyan gráfhalmazt értünk, amely zárt az izomorfiára. Azaz T gráftulajdonság, ha minden $G = (V, E)$ gráf és π permutáció esetén $G \in T \Leftrightarrow \pi(G) \in T$ (ahol $\pi(G) = (V, \{\{\pi(u), \pi(v)\} : \{u, v\} \in E\})$). Sokszor előnyös, ha egy N elemű csúcshalmaz esetén a csúcsokat számoknak, konkrétan az első N darab pozitív egésznek feleltetjük meg, vagyis $V = [N]$.

A T gráftulajdonságot tesztelő algoritmust *T-tesztelő*nek nevezzük. Ez minden esetben olyan randomizált algoritmus lesz, amelynek orákulum-hozzáférése van a G gráfhoz, és meg kell állapítania, hogy az teljesül-e, hogy $G \in T$, vagy az, hogy G ϵ-távol van T-től. Az orákulumhoz való hozzáférés módja és a távolságmérték a lényegi különbségek az egyes modellek között.

Sok T-tesztelő algoritmus úgy működik, hogy vesz egy véletlen csúcshalmazt, amit *minta*nak nevezünk, és azt ellenőrzi, hogy a minta által feszített részgráf teljesít-e egy T' tulajdonságot. Gyakran ráadásul $T' = T$.

4.1. Definíció. *Egy T-tesztelő* kanonikus*, ha egyenletesen kiválasztja m darab csúcsát a vizsgált N csúcsú gráfnak, és pontosan akkor fogad el, ha a kiválasztott csúcsok által feszített részgráf teljesít egy T' tulajdonságot. Itt T' olyan gráftulajdonság, ami függhet N-től és T-től is.*

Ismert, hogy bármely T-hez adható ilyen módon működő tesztelő:

4.1. Tétel. *Legyen T tetszőleges gráftulajdonság. Ha van olyan T-tesztelő, aminek a lekérdezésbonyolultsága $q(N, \epsilon)$, akkor létezik olyan kanonikus T-tesztelő is, ami $O(q(N,\epsilon))$ csúcsot választ ki. Továbbá, ha az eredeti tesztelőnek egyoldali hibája van, akkor a kanonikusnak is, és ekkor elegendő $2q(N, \epsilon)$ csúcsot kiválasztania.*

Ezért a kanonikus tesztelő lekérdezésbonyolultsága legfeljebb négyzetesen függhet az eredeti tesztelőétől, hiszen $m \in O(q)$ méretű minta esetén minden csúcspárt lekérdezve is csak $m^2 \in O(q^2)$ darab kérdésre van szükség. Ugyanakkor a kanonikus változat időkomplexitása jelentősen nagyobb lehet az eredetiétől.

4.1. Sűrűgráf-modell

Az ebben a modellben történő tulajdonságteszteléssel elsőként Goldreich, Goldwasser és Ron foglalkozott [GGR98].

Ebben az esetben az orákulum a szomszédsági predikátum, azaz egy szimmetrikus $g : V \times V \to \{0, 1\}$ függvény, amire $g(u, v) = 1 \Leftrightarrow \{u, v\} \in E$. A távolságmérték ezen a reprezentáción alapul: két, azonos méretű csúcshalmazon értelmezett gráf távolsága azon csúcspárok aránya az összes csúcspárhoz képest, amelyek között az egyik gráfban él fut, de a másikban nem. Tehát ha a $G = ([N], E)$ gráfot a g, a $G' = ([N], E')$-t a g' függvény reprezentálja, π pedig bármely permutáció lehet az $[N]$ halmazon, akkor G és G' távolsága

$$D_a(G, G') = \frac{\min_\pi |\{(u, v) : g(u, v) \neq g'(\pi(u), \pi(v))\}|}{N^2}.$$

Ebben a formalizmusban az, hogy a G gráf ϵ-távol van a T tulajdonságtól, azt jelenti, hogy $\forall G' \in T$ esetén $D_a(G, G') > \epsilon$. Tehát ebben a modellben egy T-tesztelőtől a következőt várjuk.

4.2. Definíció. *A sűrűgráf-modellben a T gráftulajdonságot tesztelő algoritmus megkapja bemenetként a tesztelendő G gráf csúcsainak N számát és az ϵ távolságparamétert, valamint orákulum-hozzáférése van a G szomszédsági predikátumához. A tesztelő kimenete: elfogadja vagy elutasítja G-t úgy, hogy az alábbi két feltétel teljesül.*

- *Ha $G \in T$, akkor a tesztelő legalább $2/3$ valószínűséggel elfogadja.*
- *Ha G ϵ-távol van T-től, akkor legfeljebb $1/3$ valószínűséggel fogadja el.*

Ezt a modellt szomszédsági predikátum (vagy szomszédsági mátrix) modellnek is nevezik – nyilvánvaló okból. A sűrűgráf-modell elnevezést az indokolja, hogy sűrű gráfok esetén (azaz amikor az N csúcsú gráfok nincsenek távol a K_N teljes gráftól) az előbbi D_a távolságfüggvény majdnem megegyezik a gráfokban eltérő élek és az összes él számának hányadosával. Másrészt a szomszédsági mátrix alapú tárolás sűrű gráfok esetén elterjedt, hiszen ritka gráfoknál nem jó a tárkihasználása. Ezért sűrű gráfok esetén van igazán jelentősége ennek a modellnek.

4.1.1. Néhány egyszerűen tesztelhető gráftulajdonság

Bizonyos tulajdonságokat nagyon egyszerű tesztelni. Ennek több oka lehet, az egyik ok, ha bármely gráf közel van a tulajdonsághoz. Ilyenek például az összefüggő gráfok, az Euler-körrel rendelkező gráfok, a teljes párosítással rendelkező gráfok, illetve a Hamilton-körrel rendelkező gráfok halmazai. Alább az utóbbi példáját tekintjük, a többi eset hasonló ehhez.

Ha N csúcsú a gráf, és $\epsilon \geq \frac{1}{N}$, akkor ez a D_a távolságfüggvény értelmében azt jelenti, hogy $N^2 \cdot \epsilon \geq N$ eltérés engedélyezett a két élhalmaz között. Márpedig bármely N csúcsú gráfhoz hozzá tudunk adni legfeljebb N élet (konkrétan egy Hamilton-kör éleit) úgy, hogy az eredményben legyen Hamilton-kör. Tehát ekkor az algoritmus triviális eredményt adhat: minden gráfot elfogad. Mivel minden gráf ϵ-közel van T-hez, ez megfelel a feltételeinknek.

Ellenben ha $\epsilon \leq \frac{1}{N}$, akkor a tesztelő algoritmus megvizsgálja mind az N^2 csúcspárt, és ez alapján ad teljesen korrekt választ. Mivel $N^2 \leq \frac{1}{\epsilon^2}$, ez $1/\epsilon$-ban polinomiális sok lekérdezés, és a lekérdezésbonyolultság független a gráf méretétől. (Ugyanakkor időkomplexitás

szempontjából még ellenőrizni kell, hogy a lekérdezett élek között van-e N olyan, amelyek egy Hamilton-kör élei, így a gráf méretében exponenciális a lépésszám.)

Vegyük észre, hogy ez az algoritmus a tulajdonságot teljesítő gráfokat biztosan elfogadja, azaz egyoldali hibája van.

További oka lehet az egyszerű tesztelhetőségnek, ha a tulajdonságot csak ritka gráfok teljesítik. Ebben az esetben az algoritmus néhány random csúcs közül csúcspárokat vizsgál, és akkor fogad el, ha így kevés élet talál. Ilyen tulajdonság például, hogy a gráf fa, erdő, körmentes, illetve síkba rajzolható. (A részletekért lásd [GGR98, Proposition 10.2.1.2.].)

Hasonlóan egyszerű algoritmus adható a reguláris gráfok esetére is: néhány véletlenszerűen választott csúcsot megvizsgál, hogy néhány másik random csúcs közül hánnyal szomszédos. Ha a vizsgált csúcsoknak az ez alapján tapasztalt foka csak kicsivel tér el egymástól, akkor elfogadja a gráfot. (A részletekért lásd [GGR98, Proposition 10.2.1.3.].)

4.1.2. A k-színezhetőség tesztelése

4.3. Definíció. *Egy $G = (V, E)$ gráf k-színezhető, ha adható olyan $c : V \to [k]$ színfüggvény, amelyre minden $u, v \in V$ esetén teljesül, hogy ha $\{u, v\} \in E$, akkor $c(u) \neq c(v)$.*

Az, hogy egy gráf ϵ-távol van ettől a tulajdonságtól, azt jelenti, hogy bárhogyan színezzük a gráfot k színnel, több mint ϵN^2 olyan csúcspár lesz, amely csúcsai azonos színűek, és él fut közöttük.

Az algoritmus egy egyszerű kanonikus tesztelő, ami véletlenszerűen kiválasztja a gráf egy kis méretű X részhalmazát, és akkor fogad el, ha az X minta által feszített részgráf k-színezhető. Ez alapján látható, hogy az algoritmusnak egyoldali hibája van, és minden elutasított bemenethez mutatni is tud egy kis méretű ellenpéldát, azaz olyan részgráfot, ami nem k-színezhető.

Bemenet:
N: a gráf csúcsainak száma (feltesszük, hogy a csúcshalmaz $V = [N]$);
ϵ: távolságparaméter;
k: a színek száma;
$g : V \times V \to \{0, 1\}$: hozzáférés a G gráf szomszédság-orákulumához.

A fő lépések:

1. Válasszunk egyenletes eloszlással az $[N]$ halmazból egy $O(k^2 \log(k)/\epsilon^3)$ méretű X mintát.
2. Kérdezzük le minden $(u, v) \in X \times X$ csúcspárra $g(u, v)$ értékét. Ezzel megismertük G-nek az X által feszített részgráfját, G_X-et.
3. **Ha** G_X k-színezhető, **akkor** fogadjunk el; **egyébként** utasítsunk el.

4.1. Algoritmus: A k-színezhetőséget tesztelő algoritmus.

Az algoritmus 3. lépésében annak ellenőrzése, hogy G_X k-színezhető-e, például a nyilvánvaló, exponenciális lépésszámú algoritmussal történhet (minden lehetőség kipróbálása nyers erővel $O(k^{|X|})$ lépés). Ez ugyanis a lekérdezésbonyolultságon nem ront, az időbonyolultság pedig a probléma **NP**-teljessége miatt jelen tudásunk szerint mindenképp exponenciális – kivéve, ha $k = 2$. Ez utóbbi esettel a következő szakaszban részletesen foglalkozunk.

Ha a G gráf nem k-színezhető, és ezt az algoritmus kideríti, akkor G_X k-színezhetőségének ellenőrzése ellenpéldát is szolgáltat. Annak bizonyításától eltekintünk, hogy ez az algoritmus valóban teljesíti a tulajdonságtesztelésre vonatkozó feltételeinket, helyette a következő szakaszban, a párosság tesztelésénél végzünk el egy hasonló analízist.

4.1.3. Gráfok párosságának tesztelése

Bár a k-színezhetőség $k = 2$ eseteként már megoldottuk a problémát, azért vizsgáljuk külön a 2-színezhetőséget, mert így hatékonyabb algoritmust kaphatunk, mintha az általános változatot használnánk.

Ez az algoritmus lényegében ugyanúgy működik, mint a k-színezhetőség tesztelése. Véletlenszerűen mintavételezünk egy kis méretű csúcshalmazt a gráf csúcsai közül, és pontosan akkor fogadunk el, ha a minta által feszített részgráf páros.

Bemenet:
N: a gráf csúcsainak száma (feltesszük, hogy a csúcshalmaz $V = [N]$);
ϵ: távolságparaméter;
$g : V \times V \to \{0, 1\}$: hozzáférés a G gráf szomszédság-orákulumához.

A fő lépések:

1. Válasszunk egyenletes eloszlással az $[N]$ halmazból egy $O(\log(1/\epsilon)/\epsilon^2)$ méretű X mintát.
2. Kérdezzük le minden $(u, v) \in X \times X$ csúcspárra $g(u, v)$ értékét. Ezzel megismertük G-nek az X által feszített részgráfját, G_X-et.
3. **Ha** G_X páros, **akkor** fogadjunk el; **egyébként** utasítsunk el.

4.2. Algoritmus: A párosságot tesztelő algoritmus.

Az algoritmus 3. lépésében a G_X gráf párosságának ellenőrzése történhet a szokásos módon, a BFS (szélességi bejárás) algoritmussal. A lekérdezéskomplexitás polinomiális $1/\epsilon$-ban, illetve a futásidő is, hiszen a mintavételezés és a lekérdezések után már csak egy $1/\epsilon$-ban polinomiális méretű gráfon kell BFS-t futtatni.

4.4. Definíció. *A $G = (V, E)$ gráfban egy $\{u, v\} \in E$* sértő él *a csúcsok $\{V_1, V_2\}$ partíciójára nézve, ha mindkét végpontja az egyik halmazba esik, azaz $u, v \in V_1$ vagy $u, v \in V_2$. Ha a $\{V_1, V_2\}$ partícióra nézve legfeljebb ϵN^2 sértő él van, akkor ezt a partíciót* ϵ-jónak, *egyébként* ϵ-rossznak nevezzük. *A sértő élektől mentes partíciót* tökéletesnek hívjuk.

Jelölés: Egy $G = (V, E)$ gráfban egy $v \in V$ csúcs szomszédainak halmazát $\Gamma(v)$-vel jelöljük. Azaz $\Gamma(v) = \{u \in V : \{u, v\} \in E\}$. Egy $X \subseteq V$ csúcshalmaz esetén pedig $\Gamma(X) = \cup_{v \in X} \Gamma(v)$.

4.2. Tétel. *A 4.2. algoritmus tulajdonságtesztelő a páros gráf tulajdonságra. Lekérdezés- és időkomplexitása is polinomiális $1/\epsilon$-ban. Az algoritmusnak egyoldali hibája van, azaz minden páros gráfot elfogad.*

4.2.1. Megjegyzés. *Az alábbinál bonyolultabb módon az is belátható, hogy $\tilde{O}(1/\epsilon)$ csúcs mintavételezése is elegendő, ami $\tilde{O}(1/\epsilon^2)$ lekérdezést jelent [AK02]. Azt is belátták már, hogy egy nem adaptív párosságtesztelő algoritmusnak mindenképp szükséges legalább $\Omega(1/\epsilon^2)$ kérdést feltennie [BT04].*

Bizonyítás. A tételnek a polinomialitásra vonatkozó állítását lényegében már láttuk: az X minta által feszített $O(\log(1/\epsilon)/\epsilon^2)$ méretű gráf minden csúcspárjának lekérdezése $O(\log^2(1/\epsilon)/\epsilon^4)$ lekérdezést igényel, így a lekérdezésbonyolultság rendben van. A mintavételezés, a lekérdezések és a szélességi bejárás időigénye mind polinomiális $1/\epsilon$-ban: $O(\log(1/\epsilon)/\epsilon^2) + O(\log^2(1/\epsilon)/\epsilon^4) + O(\log^2(1/\epsilon)/\epsilon^4) = O(\log^2(1/\epsilon)/\epsilon^4)$ a lépésszám.

Az egyoldali hiba is világos: ha G páros gráf, akkor minden részgráfja is páros, tehát nem tudunk olyan mintát választani, amely által feszített részgráf nem páros. Így az algoritmus biztosan elfogadó választ ad. Ha pedig elutasít, azt azért teszi, mert G_X nem

páros, azaz talált egy kis méretű ellenpéldát (páratlan hosszú kört), amivel bizonyítani lehet a nem-párosságot.

Tehát csak azt kell belátnunk, hogy a 4.2. algoritmusra teljesül, hogy ha G ϵ-távol van a páros gráfoktól, akkor legfeljebb $1/3$ valószínűséggel fogadja el. Ezzel ekvivalens: ha G ϵ-távol van a páros gráfoktól, akkor annak a valószínűsége, hogy G_X páros, legfeljebb $1/3$.

A mintavételezéssel kapott X csúcshalmazt két diszjunkt halmaz, U és S uniójaként tekintjük, ahol $|U| = t \in \Theta(\log(1/\epsilon)/\epsilon)$, és $|S| = m \in \Theta(t/\epsilon)$ – ezzel $|X| \in O(\log(1/\epsilon)/\epsilon^2)$ teljesül, amint az algoritmusban szerepel. Az a terv, hogy U-nak egy $\{U_1, U_2\}$ partíciója alapján a teljes V csúcshalmaznak megkapjuk egy partícióját. S szerepe az lesz, hogy ellenőrizze V-nek az így kapott partícióját. U-nak adott $\{U_1, U_2\}$ partíciójából úgy kapjuk meg V-nek a $\{V_1, V_2\}$ partícióját, hogy $V_1 = \Gamma(U_2)$, és $V_2 = V \setminus V_1$.

Tehát azok a csúcsok, amelyeknek van U_2-beli szomszédjuk, V_1-be kerülnek; a többi csúcs pedig – tehát azok, amelyeknek van U_1-beli szomszédjuk, de nincs U_2-beli, valamint azok, amelyeknek egyáltalán nincs U-beli szomszédjuk – V_2-be. Vegyük észre, hogy azok a csúcsok, amelyeknek nincs U-beli szomszédjuk, bármelyik oldalra kerülhetnének az $\{U_1, U_2\}$ partíció alapján. Ezért szeretnénk biztosítani, hogy a legtöbb csúcsnak (de legalábbis a „fontos" csúcsoknak) legyen U-beli szomszédja.

4.5. Definíció. *Egy $v \in V$* fontos csúcs*, ha a fokára teljesül $d(v) \geq \frac{\epsilon}{3}N$.*
Egy $U \subseteq V$ csúcshalmaz korrekt halmaz*, ha a fontos csúcsok közül legfeljebb $\frac{\epsilon}{3}N$ olyan van, aminek nincs U-beli szomszédja.*

Látható, hogy az előbbi definíció nem követeli meg, hogy egy korrekt halmazban minden fontos csúcsnak legyen szomszédja, csak azt, hogy majdnem mindegyiknek. Ennek az engedékenységnek a következménye, hogy az algoritmusunkban a ténylegesen vizsgált G_X részgráf mérete független lehet a G gráf méretétől.

4.3. Állítás. *Egy egyenletes eloszlással választott $t \in \Omega(\log(1/\epsilon)/\epsilon)$ méretű $U \subseteq V$ csúcshalmaz legalább 5/6 valószínűséggel korrekt.*

Bizonyítás. Legyen $v \in V$ tetszőleges fontos csúcs. Annak a valószínűsége, hogy v-nek nincs szomszédja az egyenletesen választott t méretű U-ban, legfeljebb $(1 - \epsilon/3)^t$, hiszen bármely U-beli csúccsal legalább $\epsilon/3$ valószínűséggel szomszédos a fontossága miatt. Tovább becsülve felülről

$$\left(1 - \frac{\epsilon}{3}\right)^t \leq e^{-t\frac{\epsilon}{3}} = \frac{\epsilon}{18}.$$

Az egyenlőtlenség indoklása: $(1 - \frac{\epsilon}{3})^t = (1 - \frac{t\epsilon/3}{t})^t$, amiből $\lim_{n\to\infty}(1 - \frac{x}{n})^n = e^{-x}$ miatt, és azért, mert $(1 - \frac{x}{n})^n$ monoton nő, következik az egyenlőtlenség. Az egyenlőség pedig azért igaz, mert $t = 3\log(18/\epsilon)/\epsilon$ választással $t \in \Theta(\log(1/\epsilon)/\epsilon)$ teljesül, ezt a t értéket behelyettesítve pedig épp az eredményt kapjuk.

Amit beláttunk: $p = \Pr[\text{egy adott fontos csúcsnak nincs } U\text{-beli szomszédja}] \leq \epsilon/18$. Jelölje az Y valószínűségi változó azon fontos csúcsok számát, amelyeknek nincs U-beli szomszédjuk. Mivel Y binomiális eloszlású, tetszőleges elemre a feltétel teljesülésének esélye $p \leq \epsilon/18$, és legfeljebb N az elemek (fontos csúcsok) száma, a várható értékre $E(Y) \leq N\frac{\epsilon}{18}$ teljesül. A Markov-egyenlőtlenség miatt $\Pr\left[Y > N\frac{\epsilon}{3}\right] \leq \frac{E(Y)}{N\epsilon/3} \leq 1/6$.

Azt láttuk be, hogy annak a valószínűsége, hogy legalább $N\frac{\epsilon}{3}$ fontos csúcsnak nincs U-beli szomszédja, legfeljebb $1/6$. Ez a korrekt halmazok definíciója miatt éppen azt jelenti, hogy U legalább $5/6$ valószínűséggel korrekt. □

4.6. Definíció. *Azt mondjuk, hogy az $\{u, v\} \in E$ él* zavarja *az (U_1, U_2) partíciót, ha mindkét végpontja beleesik valamelyik U_i ($i \in \{1, 2\}$) szomszédságába, azaz $u, v \in \Gamma(U_1)$ vagy $u, v \in \Gamma(U_2)$ teljesül.*

Egy (U_1, U_2) partíciót ϵ-hasznos*nak nevezünk, ha azt kevesebb mint $\frac{\epsilon}{3}N^2$ él zavarja, egyébként ϵ-haszontalannak hívjuk.*

4.4. Állítás. *Ha a $G = (V, E)$ gráf ϵ-távol van a párosságtól, akkor bármely $U \subseteq V$ korrekt csúcshalmaznak bármely partíciója ϵ-haszontalan.*

Bizonyítás. Az, hogy G ϵ-távol van a párosságtól, azt jelenti, hogy a csúcshalmaz bármely $V = \{V_1, V_2\}$ partíciójára legalább ϵN^2 olyan él van, ami sérti a partíciót, azaz két V_1-beli vagy két V_2-beli csúcs között fut.

Becsüljük felülről azon élek számát, amelyek sértik ugyan az $U = \{U_1, U_2\}$ partícióból származó $V = \{V_1, V_2\}$ partíciót, de ez nem derül ki számunkra, ha csak $\{U_1, U_2\}$-t tekintjük. Tehát azokról az élekről van szó, amelyek sértik $\{V_1, V_2\}$-t, de nem zavarják $\{U_1, U_2\}$-t. Nevezzük ezeket SNZ (sértő, de nem zavaró) éleknek.

Azok az élek, amelyek mindkét végpontja $\Gamma(U)$-beli, nem SNZ-k, hiszen azok vagy nem sértik $\{V_1, V_2\}$-t (ha $\Gamma(U_1)$ és $\Gamma(U_2)$ között futnak), vagy zavarják $\{U_1, U_2\}$-t (ha mindkét végpontjuk ugyanabban a $\Gamma(U_i)$-ban van, $i \in \{1, 2\}$). Tehát azon élek száma, amelyeknek legalább az egyik végpontja $V \setminus \Gamma(U)$-beli, felülről becsli az SNZ élek számát.

Ezt tovább becsülhetjük felülről. Ehhez bontsuk ketté ezt az élhalmazt:

- Azok az élek, amelyek $V \setminus \Gamma(U)$-beli fontos csúcsra illeszkednek. Mivel U korrekt halmaz, legfeljebb $\frac{\epsilon}{3}N$ darab $\Gamma(U)$-n kívüli fontos csúcs van, és ezek mindegyikének legfeljebb N a foka, ezért maximum $\frac{\epsilon}{3}N^2$ ilyen él lehet.
- Azok az élek, amelyek $V \setminus \Gamma(U)$-beli nem fontos csúcsra illeszkednek. Ezek számát rögtön tovább becsüljük felülről azzal, hogy az összes nem fontos csúcsra illeszkedő élet tekintjük. Nem fontos csúcsból legfeljebb N darab lehet, és a fontos csúcsok definíciója miatt mindegyik nem fontos csúcsnak kevesebb mint $\frac{\epsilon}{3}N$ szomszédja van. Ezért ilyen élből szintén maximum $\frac{\epsilon}{3}N^2$ darab lehet.

Azt kaptuk, hogy az SNZ élek száma felülről becsülhető ezen két érték összegével, $\frac{2\epsilon}{3}N^2$-tel. Mivel az összes $\{V_1, V_2\}$-t sértő él száma legalább ϵN^2, következik, hogy ezek között legalább $\frac{\epsilon}{3}N^2$ olyan él, van, ami zavarja $\{U_1, U_2\}$-t. Tehát az $\{U_1, U_2\}$ partíció ϵ-haszontalan. □

A következő állításból az fog kiderülni, hogy ha egy $\{U_1, U_2\}$ partíció ϵ-haszontalan, akkor erről nagy valószínűséggel meggyőződhetünk az S minta vizsgálatával. A bizonyítékot egy olyan $(v, v') \in S \times S$ szomszédos csúcspár szolgáltatja, ahol v és v' is ugyanabban a $\Gamma(U_i)$-ben van, mondjuk $\Gamma(U_1)$-ben. Tehát mindkét csúcsnak van U_1-beli szomszédja, legyen v szomszédja u, v'-é pedig u', ahol $u, u' \in U_1$. Ha esetleg $u = u'$, akkor $\{v, v', u\}$ háromszöget alkot a G_X részgráfban: ezt észreveszi az algoritmusunk. Ha $u \neq u'$, akkor a v és v' közötti él miatt u és u' mindenképp különböző oldalra kerül (u oda, ahova v', u' oda, ahova v) az $X = U \cup S$ halmaz bármely tökéletes partíciója esetén. Azaz, mivel $u, u' \in U_1$, az $\{U_1, U_2\}$ partíció nem egészíthető ki X-nek egy tökéletes partíciójává, tehát az $\{U_1, U_2\}$ alapján particionált csúcshalmaz esetén G_X ellenőrzésekor az algoritmusunk talál páratlan kört.

4.5. Állítás. *Legyen $U \subseteq V$ egy $|U| = t$ méretű csúcshalmaz, és tekintsük ennek egy rögzített ϵ-haszontalan $\{U_1, U_2\}$ partícióját. Ekkor bármely, egyenletes eloszlással választott $m \in \Omega(t/\epsilon)$ méretű $S \subset V$ csúcshalmazra*

$$\Pr_S[\forall (v, v') \in S \times S, i \in \{1, 2\} : (v, v') \notin E(\Gamma(U_i), \Gamma(U_i))] < \frac{1}{3}2^{-(t+1)}.$$

Ahol $(v, v') \in E(A, B)$ jelentése egy $G = (V, E)$ gráf és $A, B \subseteq V$ esetén, hogy $v \in A$, $v' \in B$ és $\{v, v'\} \in E$. A $(v, v') \notin E(A, B)$ pedig ennek a negáltja. Tehát az állítás szerint

egy ϵ-haszontalan $\{U_1, U_2\}$ partíció esetén nagy valószínűséggel van olyan él S-en belül, ami zavarja ezt a partíciót. Tehát az $\{U_1, U_2\}$ nem egészíthető ki $X = U \cup S$-nek egy tökéletes partíciójává.

Bizonyítás. Mivel $\{U_1, U_2\}$ ϵ-haszontalan, legalább $\frac{\epsilon}{3}N^2$ él zavarja, vagyis

$$\Pr_{v,v'}[\exists i \in \{1,2\} : (v,v') \in E(\Gamma(U_i), \Gamma(U_i))] \geq \frac{\epsilon}{3}.$$

Az m elemű S halmazt egyenletesen választjuk, ami tekinthető úgy, hogy $m/2$ darab (v, v') csúcspárt választunk függetlenül, egyenletes eloszlással. Ahhoz az eseményhez, aminek a valószínűsége szerepel az állítás bal oldalán, az kell, hogy ezek egyike se zavarja a partíciót. Ezért

$$\Pr_S[\forall (v,v') \in S \times S, i \in \{1,2\} : (v,v') \notin E(\Gamma(U_i), \Gamma(U_i))] \leq \left(1 - \frac{\epsilon}{3}\right)^{m/2}.$$

A jobb oldalt tovább becsüljük felülről, először a 4.3. állítás bizonyításában már látott módon, ϵ-ban exponenciális függvénnyel. Legyen $m = \frac{6}{\epsilon}\left((t+1)\ln(2) + \ln(3)\right)$, erre teljesül, hogy $m \in \Theta(t/\epsilon)$. Ezt behelyettesítve kapjuk a bizonyítandó állítást.

$$\left(1 - \frac{\epsilon}{3}\right)^{m/2} \leq e^{-m\epsilon/6} = \frac{1}{3}2^{-(t+1)}.$$

□

A 4.2. tétel bizonyításának befejezéséhez vegyük észre, hogy G_X csak akkor lehet páros gráf, ha (a) U nem korrekt, vagy ha (b) U ugyan korrekt, de van olyan partíciója, hogy egy olyan él sincs G_X-ben, ami zavarja a partíciót. Az (a) eset valószínűsége a 4.3. állítás miatt legfeljebb 1/6.

A (b) esetben U korrekt, ezért a 4.4. állítás miatt U bármely partíciója ϵ-haszontalan. A 4.5. állítás szerint ebből következik, hogy bármely U-partíció esetén annak a valószínűsége, hogy S éleinek vizsgálatával nem találunk rossz élet, legfeljebb $\frac{1}{3}2^{-(t+1)}$. Tehát annak az esélye, hogy U-nak a lehetséges $2^{|U|} = 2^t$ darab partíciója közül van olyan, amelyik esetén nem találunk S-ben rossz élet, a Boole-egyenlőtlenség miatt legfeljebb $2^t \cdot \frac{1}{3}2^{-(t+1)} = \frac{1}{6}$.

Ezzel beláttuk, hogy ha G ϵ-távol van a párosságtól, akkor legfeljebb $1/6 + 1/6 = 1/3$ valószínűséggel páros a G_X részgráf.

□

A gráfok párosságát tesztelő algoritmust megvalósítottunk, és a futtatások során szerzett tapasztalatokat értelmeztük. Ezek az eredmények a 7.2. szakaszban olvashatók.

4.2. Egyéb gráfmodellek

Két további modellről teszünk említést. Ezek esetén is áttekintjük a párosságtesztelés példáját, de már részletes elemzés nélkül. Forrásként [Gol17] 9. és 10. fejezetét használtuk.

4.2.1. Korlátos fokszámú gráfok modellje

Ez a modell Goldreich és Ron nevéhez fűződik [GR04].

A sűrűgráf-modell esetén a szomszédsági mátrix megadáshoz hasonló volt a hozzáférés a gráfhoz. Ebben a modellben egy másik elterjedt gráfmegadási móddal, a szomszédsági listával van analógia. Ehhez fel kell tennünk, hogy van egy d felső korlát a G-beli csúcsok fokszámaira. Ekkor a $G = (V, E)$ gráfra az orákulum egy $g : V \times [d] \to V \cup \{0\}$ függvény,

ahol $g(u,i) = v$, ha az $u \in V$ csúcs i-edik szomszédja (a szomszédsági listában) a $v \in V$ csúcs; és $g(u,i) = 0$, ha u-nak kevesebb mint i darab szomszédja van. Irányítatlan gráfról lévén szó, feltehetjük, hogy $g(u,i) = v \Leftrightarrow \exists j \in [d] : g(v,j) = u$.

A reprezentáció itt is meghatározza a távolságmértéket: azon (u,i) pároknak az összes ilyen párhoz vett aránya, amelyek eltérő értéket adnak a gráfokat reprezentáló függvényekbe helyettesítve. Formálisan: ha a $G = ([N], E)$ gráfot a g, a $G' = ([N], E')$-t a g' függvény reprezentálja, π pedig bármely permutáció lehet az $[N]$ halmazon, akkor G és G' távolsága

$$D_b(G, G') = \frac{\min_\pi \sum_{u \in [N]} |\{v : \exists i \; g(u,i) = v\} \triangle \{v : \exists i \; g'(\pi(u), i) = \pi(v)\}|}{d \cdot N},$$

ahol $\triangle$ a halmazok szimmetrikus differenciáját jelöli, azaz $A \triangle B = (A \cup B) \setminus (A \cap B)$.

Az, hogy a G gráf ϵ-távol van a T tulajdonságtól, ebben a modellben azt jelenti, hogy $\forall G' \in T$ esetén $D_b(G, G') > \epsilon$.

A korlátos fokszám elnevezés világos: csak olyan gráfok reprezentálhatók így, amelyek maximális fokszáma legfeljebb d.

A párosság tesztelése

Belátható, hogy a korlátos fokszámú gráfok modellje esetén mindenképp szükséges $\Omega(\sqrt{N})$ kérdés a párosság teszteléséhez [GR04] – ellentétben a sűrűgráf-modellel, ahol N-től független darabszámú és $1/\epsilon$-ban polinomiálisan sok kérdéssel tesztelhető ez a tulajdonság. Ez a korlát lényegében szoros, ugyanis $\tilde{O}(\sqrt{N}) \cdot \mathrm{poly}(1/\epsilon)$ kérdés elegendő [GR99].

A tesztelő algoritmus véletlenszerűen kiválasztott csúcsból induló véletlen bolyongásokat (random walk) végez a gráfon, és akkor fogad el, ha az eközben felfedezett részgráf páros. Ezt mutatja be a 4.3. algoritmus.

Bemenet:
N: a gráf csúcsainak száma (feltesszük, hogy a csúcshalmaz $V = [N]$);
d fokszámkorlát;
ϵ: távolságparaméter;
$g : V \times [d] \to V \cup \{0\}$: hozzáférés a G gráf szomszédsági lista orákulumához.

A fő lépések:

1. **Ciklus** $t = \Theta(1/\epsilon)$ körön át
   - 1.1. Válasszunk egyenletes eloszlással egy $s \in [N]$ csúcsot.
   - 1.2. Végezzünk $K = \sqrt{N} \cdot \mathrm{poly}(\log(N)/\epsilon)$ darab véletlen bolyongást s-ből indulva, mindegyik hossza legyen $\ell = \mathrm{poly}(\log(N)/\epsilon)$.
   - 1.3. Jelölje R_0 azon csúcsok halmazát, amelyeket a bolyongások bármelyikében páros sok lépésben értünk el s-ből; R_1 pedig ugyanezt páratlan sok lépésre.
   - 1.4. **Ha** $R_0 \cap R_1 \neq \emptyset$, **akkor** utasítsunk el.
2. Fogadjunk el.

4.3. Algoritmus: A párosság tesztelése korlátos fokszámú esetben.

Véletlen bolyongás alatt azt értjük, hogy egy kezdő csúcsból indulunk, és minden lépésben az aktuális csúcs szomszédai közül egyenletes eloszlással választjuk ki a következő csúcsot. Így végül egy sétát kapunk a gráfban. A bolyongáshoz minden lépésben tudni kell az aktuális csúcs fokszámát. Ez ebben a modellben bináris kereséssel $\log d$ kérdéssel egyszerűen meghatározható egy-egy csúcsra. Így, felhasználva, hogy $d < N$, az algoritmus lekérdezéseinek száma $t \cdot K \cdot \ell \cdot \log d \in O(\sqrt{N}) \cdot \mathrm{poly}(\log(N)/\epsilon) = \tilde{O}(\sqrt{N}) \cdot \mathrm{poly}(1/\epsilon)$.

Az világos, hogy az algoritmus minden páros gráfot 1 valószínűséggel elfogad, vagyis egyoldali hibája van. Az is belátható, hogy a párosságtól ϵ-távoli gráfokat legalább $2/3$ valószínűséggel elutasítja, azaz a 4.3. algoritmus valóban tulajdonságtesztelő.

4.2.2. Általános gráfmodell

Ezt a modellt Parnas és Ron vezette be [PR02].

Ebben az esetben a gráfot reprezentáló függvény nem rögzített, így a távolságmértéket sem lehet ehhez kötni, az független a reprezentációtól. A lekérdezések általában az előző két modell bármelyikének megfelelően történhetnek, azaz a szomszédsági mátrix és a szomszédsági lista orákulumától is lehet kérdezni.

A távolságmérték azt adja meg, hogy a két gráf élhalmaza hány elemben tér el egymástól, osztva az összes él számával – a két gráf közül a nagyobb élhalmaz méretével. Tehát ha $G = ([N], E)$ az egyik, $G' = ([N], E')$ a másik gráf, π pedig bármely permutáció lehet az $[N]$ halmazon, akkor G és G' távolsága

$$D_g(G, G') = \frac{\min_\pi |E \triangle \{\{\pi(u), \pi(v)\} : \{u, v\} \in E'\}|}{\max(|E|, |E'|)}.$$

Az általános gráfmodell elnevezés helytálló, mivel ez a legáltalánosabb modell a három közül: nem feltételezi sem azt, hogy sűrű legyen a gráf, sem azt, hogy korlátozott fokszámú legyen. Épp ezért ennek a modellnek a vizsgálata a legösszetettebb feladat.

A párosság tesztelése

Az általános gráfmodellben a párosság teszteléséről Kaufman, Krivelevich és Ron írtak cikket [KKR04]. Az általuk leírt algoritmus alapvetően a korlátos fokszámú eset algoritmusát hajtja végre, így ahhoz hasonló korlátokat kaptak. Jelölje $m = |E|$ a vizsgált G gráf éleinek számát. A cikk fő eredménye, hogy egyrészt szükséges legalább $\Omega(\min\{\sqrt{N}, N^2/m\})$ kérdés, másrészt elegendő $O(\min\{\sqrt{N}, N^2/m\}) \cdot \mathrm{poly}(\log(N)/\epsilon)$ darab.

Az utóbbi eredményt megvalósító algoritmushoz először meg kell becsülni a gráf átlagos fokszámát, amit $\bar{d}$-vel jelölünk. Ez – itt nem részletezett módon – $\tilde{O}(\sqrt{N})$ kérdéssel megtehető. Vegyük észre, hogy $m = N\bar{d}/2$, tehát a lekérdezésbonyolultságban szereplő $O(\min\{\sqrt{N}, N^2/m\})$ kifejezés helyett $O(\min\{\sqrt{N}, N/\bar{d}\})$ is írható.

Ahhoz, hogy a korlátos fokszámú gráfokra vonatkozó algoritmust használhassuk, korlátos fokszámúvá kell tenni a bemeneti általános G gráfot. Ezért egy visszavezetést adunk: G-hez hozzárendelünk egy G' korlátos fokszámú gráfot, aminek a párossága, illetve a párosságtól való ϵ-távolsága alapján G-re is hasonló következtetést vonhatunk le. Belátható, hogy ez megtehető olyan módon, hogy G' maximális foka legfeljebb $2\bar{d}$, és benne a csúcsok és az élek száma sem sokkal nagyobb, mint G-ben. A $G \mapsto G'$ leképezés részleteit nem ismertetjük, de a lényege, hogy G-nek a $\bar{d}$-nél magasabb fokú csúcsait egy-egy $\bar{d}$-reguláris páros gráffal helyettesítjük.

A továbbiakban tehát G'-vel dolgozunk: belátható, hogy a G orákulumaihoz való hozzáféréssel ez emulálható. Bár G' paraméterei eltérhetnek G-éitől, a nagyságrendjük ugyanaz, így nem vezetünk be új jelöléseket. Két esetet különböztetünk meg, mindkettőben lényegében a 4.3. algoritmust hajtjuk végre.

- Ha $\bar{d} \leq \sqrt{N}$, akkor a 4.3. algoritmust az eredeti formájában futtatjuk. Erről a korlátos fokszámú esetnél már láttuk, hogy $O(\sqrt{N}) \cdot \mathrm{poly}(\log(N)/\epsilon)$ kérdést igényel.
- Ha $\bar{d} > \sqrt{N}$, akkor $K = \mathrm{poly}(\log(N)/\epsilon) \cdot \sqrt{N/\bar{d}}$ és $\ell = \mathrm{poly}(\log(N)/\epsilon)$ paraméterekkel hajtjuk végre a 4.3. algoritmust. További különbség, hogy ez esetben az

algoritmus 1.4. lépése helyett a szomszédsági mátrix orákulumhoz intézett kérdésekkel ellenőrizzük a párosságot. Minden R_0-beli csúcspárra megnézzük, hogy fut-e közöttük él, és ha igen, akkor elutasítunk; majd ugyanezt megismételjük R_1-re is. Ez összesen $t \cdot (K \cdot \ell \cdot \log d + (K\ell)^2) \in O(N/\bar{d}) \cdot \mathrm{poly}(\log(N)/\epsilon)$ kérdés.

Ezzel beláttuk az eljárás komplexitására vonatkozó állítást. Erről az algoritmusról is belátható, hogy tulajdonságtesztelő, az egyoldali hiba pedig itt is nyilvánvaló. Vegyük észre, hogy az első esetben, azaz ha $\bar{d} \leq \sqrt{N}$, akkor $O(\sqrt{N}) \subseteq O(N/\bar{d})$; míg a második esetben épp fordított a helyzet. Tehát ez az eljárás hatékonyan általánosítja a 4.3. algoritmust.

5. fejezet

Feszített-C_4-mentesség tesztelése

A Szemerédi-féle regularitási lemma [Sze78] erős gráfelméleti eszköz, sok általános eredményt sikerült már bizonyítani a segítségével. Az egyik ilyen tétel szerint tetszőleges rögzített H gráf esetén tesztelhető a bemeneti gráf $|V(G)| = N$ méretétől független kérdésszámmal, hogy a G gráfra igaz-e, hogy nem tartalmaz H-val izomorf feszített részgráfot [AFKS00]. Ugyanakkor a hatékonyság szempontjából az is fontos kérdés, hogy hogyan függ a kérdések száma a távolságparamétertől. Ebből a szempontból már nem tekinthető hatékonynak az eredményben szereplő tesztelő algoritmus, ugyanis a lekérdezéskomplexitása a távolságparaméter reciprokának, azaz $1/\epsilon$-nak torony-típusú függvénye. Torony-típusú függvény (tower function) például $T(x)$, ahol a függvény értéke egy x magas 2-hatvány „torony", pl. $T(3) = 2^{2^2}$. (Valójában az eredeti eredmény még ennél is rosszabb, ún. wowzer-típusú volt, ezen később javítottak.)

Bizonyított, hogy néhány kivételtől (és azok komplementereitől) eltekintve semmilyen H esetén nem lehet a feszített H-mentesség tesztelésének lekérdezéskomplexitása polinomiális $1/\epsilon$-ban [AS06]. Ebben az értelemben tehát nehezen tesztelhető tulajdonságok ezek. A kivételek a 4 hosszú kör (C_4), és a legfeljebb 3 hosszú utak. A legfeljebb 3 hosszú utak esetéről kiderült, hogy polinomiális darabszámú kérdéssel tesztelhetők [AS06, AF15]. A C_4 esetében az eddigi legjobb eredmény exponenciális $1/\epsilon$-ban [GS18], azaz az általános eredménynél jobb, de polinomiálisnál rosszabb.

Az utóbbi, tehát a feszített-C_4-mentesség teszteléséről szóló bizonyítás egyes részeit ismertetjük az alábbiakban, a forrás tehát [GS18]. A cikkben a szerzők vállaltan nem törődtek pontos becslésekkel, az eredmény paramétereinek optimalizálásával. A folytatásban erre teszünk kísérletet, tehát az alábbi eredmények a cikkben találhatóktól helyenként eltérnek, azoknál élesebbek. Csak azon részeredmények bizonyítását közöljük, ahol változás történt az eredeti változathoz képest, és csak azokat a lemmákat mondjuk ki, amelyek e bizonyításokhoz szükségesek.

Az alábbiakban egy szomszédsági orákulummal adott (sűrű) G gráfot szeretnénk tesztelni feszített-C_4-mentesség szempontjából. Ehhez a fő eredmény azt mondja ki, hogy ha egy gráf ϵ-távol van a feszített-C_4-mentességtől, akkor egy $1/\epsilon$-ban exponenciális függvény reciprokával és N^4-nel arányos a benne található feszített C_4 példányok száma. Ebből valóban következik a tesztelés lekérdezéskomplexitásának exponencialitására vonatkozó állítás: ha G-ben van N^4/x példány C_4-ből (jelen esetben $x = 2^{\epsilon^{-c}}$ lesz), akkor s darab csúcsnégyest mintavételezve annak a valószínűsége, hogy egyetlen C_4 példányt sem találunk, az alábbi képlettel becsülhető.

$$\left(1 - \frac{N^4/x}{\binom{N}{4}}\right)^s \leq \left(1 - \frac{N^4/x}{N^4/24}\right)^s = \left(1 - \frac{24}{x}\right)^s$$

Ez $s = x/21$ esetén már kisebb, mint $1/3$, azaz $x/21$ csúcsnégyes mintavételezése elegendő. Ugyanakkor egy csúcsnégyes mintavételezése alatt négy, véletlenszerűen választott csúcs közül bármely kettő közötti élre vonatkozó kérdést, azaz összesen 6 kérdést értünk. Így összességében $6x/21 = \frac{2}{7}x$ kérdésről van szó. A fő tétel tehát a következő.

5.1. Tétel ([GS18] Theorem 1)**.** *Ha egy N csúcsú G gráf ϵ-távol van a feszített-C_4-mentességtől, akkor legalább $N^4/2^{\epsilon^{-c}}$ feszített C_4 példányt tartalmaz, ahol $c > 1$ abszolút konstans. Ebből következik, hogy a feszített-C_4-mentesség tesztelésének lekérdezéskomplexitása legfeljebb $\frac{2}{7}2^{\epsilon^{-c}}$.*

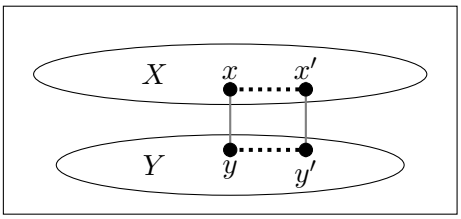
5.1.1. Megjegyzés. *Az eredeti cikkben a lekérdezéskomplexitásnak csak a nagyságrendje szerepelt, valamint a c konstans értékére rosszabb az eredmény – ezzel a bizonyítás után részletesebben foglalkozunk.*

A bizonyítás hasonló a regularitási lemmáéhoz, és több lemmán keresztül vezet. Az alapvető gondolat az, hogy belássuk, minden gráf közel van ahhoz, hogy bizonyos értelemben jól strukturált legyen. Valamivel konkrétabban: kevés élet kell módosítani ahhoz, hogy kevés csúcsot figyelmen kívül hagyva, a gráf particionálható legyen klikkekre és független halmazokra. Ráadásul az e csúcshalmazok között futó élekre is igaz, hogy közel vannak az egységességhez, tehát ahhoz, hogy vagy minden él meglegyen egy csúcshalmaz-pár között, vagy egy se legyen. Sőt, a csúcshalmazok nem túl kis méretű részhalmazaira valóban teljesül is ez az egységesség. Mindezt összekötjük a csúcshalmaz-párok közötti páros gráfok feszített kettő méretű párosításainak vizsgálatával (ami egyszerűbb kérdés) ahhoz, hogy az eredeti gráfban feszített négy hosszú köröket találjunk.

Tetszőleges $U \subseteq V(G)$ esetén $G[U]$ a G gráfnak az U csúcshalmaz által feszített részgráfját jelöli. A továbbiakban X és Y a vizsgált gráf csúcshalmazának diszjunkt részhalmazai: $X, Y \subseteq V(G)$ és $X \cap Y = \emptyset$. Az $(X, Y)_G$ páros gráfon azt a gráfot értjük, amelyet úgy kapunk, hogy a $G[X \cup Y]$ gráfból elhagyjuk azokat az éleket, amelyek mindkét végpontja X-ben, vagy mindkettő Y-ban van.

5.1. Definíció. *Egy $(X, Y)_G$ páros gráf* feszített M_2 részgráf*ján olyan x, x', y, y' csúcsnégyest értünk, ami egy kettő méretű párosítást határoz meg X és Y között, vagyis ahol $x, x' \in X$, $y, y' \in Y$, és $\{x, y\}, \{x', y'\} \in E(G)$ de $\{x, y'\}, \{x', y\} \notin E(G)$.*

A definícióhoz illusztrációként szolgál az 5.1. ábra. Vegyük észre, hogy a definíció nem mond semmit az X-en és az Y-on belüli élekről, de ha $\{x, x'\}, \{y, y'\} \in E(G)$ (ezeket az ábrán szaggatott élek jelölik), akkor a négy csúcs épp egy feszített C_4 példányt határoz meg. Az (X, Y) párra azt mondjuk, hogy *feszített-M_2-mentes*, ha $(X, Y)_G$-ben nincs feszített M_2 részgráf. Az előző észrevételhez hasonlóan látható, hogy ha X és Y klikkek, akkor (X, Y) feszített-M_2-mentessége ekvivalens $G[X \cup Y]$ feszített-C_4-mentességével.



5.1. ábra. Feszített M_2 példány X és Y között.

Az (X, Y) párt *homogén*nek nevezzük, ha az $(X, Y)_G$ páros gráf vagy üres, vagy teljes. Egy S halmaznak egy $\mathcal{P} = \{P_1, \ldots, P_r\}$ partícióját *ekvipartíció*nak nevezzük, ha bármely két elemének mérete legfeljebb eggyel tér el egymástól, azaz ha minden $i, j \in [r]$ esetén $||P_i| - |P_j|| \leq 1$.

5.2. Lemma ([GS18] Lemma 5)**.** *Ha (X, Y) feszített-M_2-mentes, akkor bármely $r \geq 1$ egész esetén van az X halmaznak olyan r elemű $X = X_1 \cup \cdots \cup X_r$ ekvipartíciója, és Y-nak olyan $r+1$ elemű $Y = Y_1 \cup \cdots \cup Y_{r+1}$ partíciója, hogy minden $i \in [r]$, $j \in [r+1]$ és $i \neq j$ esetén (X_i, Y_j) homogén.*

5.2.1. Megjegyzés. *Az egyszerűség kedvéért, követve [GS18]-at, ezentúl feltesszük, hogy az 5.2. lemma alkalmazásakor $|X|$ osztható r-rel. Ebben az esetben a kapott $\{X_1, \ldots, X_r\}$ ekvipartícióban minden elemre $|X_i| = |X|/r$ teljesül.*

Legyen adott egy halmaznak két partíciója, $\mathcal{P}_1$ és $\mathcal{P}_2$. Azt mondjuk, hogy $\mathcal{P}_2$ *finomítása* $\mathcal{P}_1$-nek, ha minden $\mathcal{P}_2$-beli halmaz részhalmaza egy $\mathcal{P}_1$-belinek, tehát $\forall A \in \mathcal{P}_2$ esetén $\exists B \in \mathcal{P}_1$, hogy $A \subseteq B$.

5.2. Definíció. *Egy N csúcsú gráf csúcshalmazának $\mathcal{P}$ partíciója* δ-homogén*, ha teljesül, hogy $\mathrm{NH}(\mathcal{P}) := \sum |U||V| \leq \delta N^2$, ahol az összeget az olyan $U, V \in \mathcal{P}$ halmazokon vesszük, ahol az (U, V) pár nem homogén.*

Ez hasonló méretű halmazok (pl. ekvipartíció) esetén azt jelenti, hogy a halmazpároknak csak legfeljebb δ része nem homogén, vagyis a legtöbb pár homogén. Arra utalva, hogy ez az érték a partíció „nemhomogenitását" méri, jelöljük $\mathrm{NH}(\mathcal{P})$-vel.

Könnyen látható, hogy ha egy $\mathcal{P}_1$ δ-homogén partíciót úgy finomít $\mathcal{P}_2$, hogy a $\mathcal{P}_1$ egyetlen eleméből a finomításnál keletkező halmazok egymás között páronként homogének legyenek[1] (azaz minden olyan $A, B \in \mathcal{P}_2$ esetén, ahol $A, B \subseteq U \in \mathcal{P}_1$, teljesül, hogy (A, B) homogén), akkor a $\mathcal{P}_2$ finomítás is δ-homogén. Mindenképp ilyen jellegű finomítás történik például akkor, ha $\mathcal{P}_1$ minden eleme klikk vagy független halmaz.

5.3. Lemma ([GS18] Lemma 6)**.** *Legyen adott egy $\delta > 0$ valós szám, valamint egy N csúcsú G gráf csúcshalmazának egy olyan $V(G) = X_1 \cup \cdots \cup X_k$ partíciója, ahol minden $i \in [k]$-ra X_i klikk, és minden $1 \leq i < j \leq k$ esetén az (X_i, X_j) pár feszített-M_2-mentes. Ekkor van olyan, az $\{X_1, \ldots, X_k\}$-t finomító δ-homogén partíció, ami legfeljebb $k(\frac{1}{\delta})^{k-1}$ elemből áll.*

5.3.1. Megjegyzés. *Az eredeti cikkben $k(2/\delta)^k$-nal becsülik fölülről a δ-homogén partíció méretét. A javításhoz mindössze valamivel pontosabb becslést végeztünk, de a teljesség kedvéért az egész bizonyítást közöljük.*

Bizonyítás. Minden (X_i, X_j) párra $(1 \leq i < j \leq k)$ alkalmazzuk az 5.2. lemmát az $r = \frac{1}{2\delta}$ paraméterrel. Így kapjuk X_i-nek a $\mathcal{P}_{i,j} = \{X_{i,j}^1, \ldots, X_{i,j}^r\}$ ekvipartícióját (ahol $\forall p \in [r] : |X_{j,i}^p| = |X_i|/r$), X_j-nek pedig a $\mathcal{P}_{j,i} = \{X_{j,i}^1, \ldots, X_{j,i}^{r+1}\}$ partícióját, ahol $(X_{i,j}^p, X_{j,i}^q)$ homogén minden $p \neq q$ esetén.

Jelöljük $\mathcal{P}_i$-vel minden $i \in [k]$ esetén az X_i-re vonatkozó $\mathcal{P}_{i,j}$ partíciók $(j \in [k], j \neq i)$ közös finomítását: kiindulunk az egyik (r vagy $r+1$ elemű) partícióból, és annak minden elemét tovább daraboljuk egy másik partíció szerint, így minden elem legfeljebb $r+1$ részre bomlik; az így kapott partíciót hasonlóan bontjuk tovább a fennmaradó $\mathcal{P}_{i,j}$ partíciókkal. Így a kapott partíció mérete $|\mathcal{P}_i| = r^{k-i}(r+1)^{i-1} \leq (r+1)^{k-1}$. Jelöljük $\mathcal{P}$-vel ezen partíciók unióját, ami a teljes csúcshalmaz eredeti $\{X_1, \ldots, X_k\}$ partícióját finomítja: $\mathcal{P} = \cup_{i \in [k]} \mathcal{P}_i$. Ez a partíció lesz az, amiről a lemma szól. $\mathcal{P}$ méretére teljesül $|\mathcal{P}| \leq k(r+1)^{k-1} \leq k(\frac{1}{\delta})^{k-1}$, a partíció méretére vonatkozó állítást tehát beláttuk.

A δ-homogenitás igazolásához $\mathrm{NH}(\mathcal{P})$-t, azaz a nemhomogén (U, V) párokon (ahol $U, V \in \mathcal{P}$) vett $\sum |U||V|$ összeget kell felülről becsülni. Nézzük meg, hogy U és V az

[1] Ez a feltétel az eredeti cikkben hiányzott, így ott pontatlanul fogalmazták meg az állítást. A cikk újabb változatában [GS19], amelyre már csak ezen fejezet jelentős részének megírása után sikerült felfigyelni, viszont már pontosítottak.

eredeti partíció melyik elemének részhalmaza: $U \subseteq X_i$ és $V \subseteq X_j$ (ahol $i \neq j$, mert minden X_i klikk és (U,V) nem homogén). Az 5.3. lemmának az (X_i, X_j) párra történő alkalmazásából keletkező partíciók valamely elemének is részhalmaza U, illetve V, azaz valamely $p \in [r]$, $q \in [r+1]$ esetén $U \subseteq X_{i,j}^p$, $V \subseteq X_{j,i}^q$. De tudjuk, hogy $(X_{i,j}^p, X_{j,i}^q)$ homogén minden $p \neq q$ esetén, tehát ha (U,V) nem homogén, akkor $p = q$.

Az alábbiakban az első lépésben az előbbi megfigyelést használjuk, tehát azt, hogy valamely $1 \leq i < j \leq k$ és $p \in [r]$ esetén $U \subseteq X_{i,j}^p$ és $V \subseteq X_{j,i}^p$. A második lépés azért jogos, mert $|X_{i,j}^p|$ az $|X_i|$ egy r elemű ekvipartíciójának eleme. Az utolsó lépés pedig egyszerűen abból következik, hogy $\sum_{p=1}^{r+1} |X_{j,i}^p| = |X_j|$.

$$\mathrm{NH}(\mathcal{P}) \leq \sum_{1 \leq i < j \leq k} \sum_{p=1}^{r} |X_{i,j}^p||X_{j,i}^p| = \sum_{1 \leq i < j \leq k} \sum_{p=1}^{r} \frac{|X_i|}{r} |X_{j,i}^p| \leq \frac{1}{r} \sum_{1 \leq i < j \leq k} |X_i||X_j|$$

Tudjuk, hogy $N^2 = (\sum_{i=1}^k |X_i|)^2 = \sum_{i=1}^k |X_i|^2 + 2\sum_{1 \leq i < j \leq k} |X_i||X_j|$, amiből következik, hogy $\sum_{1 \leq i < j \leq k} |X_i||X_j| < \frac{N^2}{2}$ (ha $k \geq 2$). Így a fenti egyenlőtlenséglánc jobb oldala tovább becsülhető felülről $\frac{N^2}{2r}$-rel, ami $r = \frac{1}{2\delta}$ miatt egyenlő δN^2-tel. Tehát a $\mathcal{P}$ partíció valóban δ-homogén. □

5.4. Lemma ([GS18] Lemma 7)**.** *Legyen adott egy $\delta > 0$ valós szám, valamint egy N csúcsú G gráf csúcshalmazának egy olyan $V(G) = X_1 \cup \cdots \cup X_k$ partíciója, ahol minden $i \in [k]$-ra X_i klikk, és minden $1 \leq i < j \leq k$ esetén az (X_i, X_j) pár feszített-M_2-mentes. Ekkor van olyan $Z \subseteq V(G)$, $|Z| < \delta N$, amire igaz, hogy a $V(G) \setminus Z$ csúcshalmaznak létezik az $\{X_1 \setminus Z, \ldots, X_k \setminus Z\}$-t finomító $\mathcal{Q} = \{Q_1, \ldots, Q_q\}$ partíciója, amire $\mathrm{NH}(\mathcal{Q}) \leq \delta N^2$. Ezenkívül vannak olyan $W_i \subseteq Q_i$ részhalmazok, amelyekre $|W_i| \geq \frac{N\delta^{4k^2-3k}}{2k^{4k}}$ minden $i \in [q]$ esetén, és (W_i, W_j) homogén minden $1 \leq i < j \leq k$ párra.*

5.4.1. Megjegyzés. *Az eredeti cikkben $|W_i| \geq N\left(\frac{\delta}{2k}\right)^{10k^2}$ szerepel. A különbség kisebb részben abból ered, hogy az 5.3. lemma javított változatát használjuk, nagyobb részben pedig abból, hogy pontosabban becslünk.*

Bizonyítás. Alkalmazzuk G-re és a csúcshalmazának a megadott $\{X_1, \ldots, X_k\}$ partíciójára az 5.3. lemmát δ paraméterrel: így kapjuk a δ-homogén $\mathcal{P}$ partíciót. Legyenek a Q_i halmazok a $\mathcal{P}$ nem túl kis méretű elemei, konkrétan $\mathcal{Q} = \{Q_1, \ldots, Q_q\} = \{U \in \mathcal{P} : |U| \geq \frac{\delta N}{|\mathcal{P}|}\}$; Z pedig a többi (tehát a kis méretű) $\mathcal{P}$-beli halmazok uniója: $Z = \cup_{U \in \mathcal{P} \setminus \mathcal{Q}}$. Így világos, hogy teljesül, hogy $\mathcal{Q}$ finomítja $\{X_1 \setminus Z, \ldots, X_k \setminus Z\}$-t, és $\mathcal{P}$ δ-homogenitása miatt az is, hogy $\mathrm{NH}(\mathcal{Q}) \leq \delta N^2$. A Z halmaz mérete is rendben van, hiszen kevesebb mint $|\mathcal{P}|$ darab kis méretű halmaz uniója, így $|Z| < |\mathcal{P}|\frac{\delta N}{|\mathcal{P}|} = \delta N$. Tehát már csak megfelelő W_i halmazokat kell találni.

Alkalmazzuk újra az 5.3. lemmát G-re és a megadott $\{X_1, \ldots, X_k\}$ partícióra, ezúttal $\delta' = \frac{\delta^2}{|\mathcal{P}|^4}$ paraméterrel: ebből legfeljebb $k(\frac{1}{\delta'})^{k-1}$ elemű δ'-homogén partíciót kapunk, amit $\mathcal{V}$-vel jelölünk. Legyen $\mathcal{W}$ a $\mathcal{P}$ és a $\mathcal{V}$ partíciók közös finomítása, ami szintén δ'-homogén, hiszen $\mathcal{V}$-t finomítja. Felső becslés adható $\mathcal{W}$ méretére:

$$|\mathcal{W}| \leq |\mathcal{P}||\mathcal{V}| \leq |\mathcal{P}|k\left(\frac{1}{\delta'}\right)^{k-1}. \tag{5.1}$$

Minden $i \in [q]$ esetén jelöljük $\mathcal{W}_i$-vel a Q_i halmaznak a $\mathcal{W}$ szerinti partícióját, azaz $\mathcal{W}_i = \{W \in \mathcal{W} : W \subseteq Q_i\}$. Minden $i \in [q]$-ra vegyünk egy elemet Q_i-ből véletlenszerűen,

egyenletes eloszlással: $w_i \in Q_i$; és legyen $W_i \in \mathcal{W}_i$ az a halmaz, amelyre $w_i \in W_i$ teljesül.

$$\Pr\left[|W_i| < \frac{|Q_i|}{2q|\mathcal{W}|}\right] < \frac{|\mathcal{W}|\frac{|Q_i|}{2q|\mathcal{W}|}}{|Q_i|} = \frac{1}{2q},$$

hiszen a legfeljebb ennyi elemű halmazok elemeinek összes száma kisebb, mint a számláló; a nevező pedig az összes elem száma, ami közül választjuk w_i-t. A Boole-egyenlőtlenség miatt annak a valószínűsége, hogy minden $i \in [q]$-ra $|W_i| \geq \frac{|Q_i|}{2q|\mathcal{W}|}$, nagyobb, mint $1 - q \cdot \frac{1}{2q} = 1/2$. Tehát 1/2-nél nagyobb valószínűséggel teljesül a következő.

$$|W_i| \geq \frac{|Q_i|}{2q|\mathcal{W}|} \geq \frac{\delta N/|\mathcal{P}|}{2|\mathcal{P}| \cdot |\mathcal{P}|k(\frac{1}{\delta'})^{k-1}} = \frac{\delta N \delta'^{k-1}}{2|\mathcal{P}|^3 k} = \frac{N\delta^{2k-1}}{2k|\mathcal{P}|^{4k-1}} \geq \frac{N\delta^{2k-1}}{2k(k\frac{1}{\delta^{k-1}})^{4k-1}} = \frac{N\delta^{4k^2-3k}}{2k^{4k}}$$

A második egyenlőtlenségben azt használtuk, hogy $|Q_i| \geq \frac{\delta N}{|\mathcal{P}|}$ ($\mathcal{Q}$-ba csak a nem túl kis méretű $\mathcal{P}$-beli halmazok kerültek), hogy $q = |\mathcal{Q}| \leq |\mathcal{P}|$, és a $|\mathcal{W}|$-re kapott (5.1) becslést. A harmadik lépésben csak átrendezés történik, a következőben pedig $\delta' = \frac{\delta^2}{|\mathcal{P}|^4}$ helyettesítés. Az utolsó előtti lépés $|\mathcal{P}|$-nek az 5.3. lemmából következő felső becslését használja, végül pedig még egy átrendezés történik.

Bármely $1 \leq i < j \leq q$ esetén annak a valószínűsége, hogy a (W_i, W_j) pár nem homogén

$$\sum_{\text{nemhomogén } (W,W') \in \mathcal{W}_i \times \mathcal{W}_j} \frac{|W||W'|}{|Q_i||Q_j|} \leq \left(\frac{|\mathcal{P}|}{\delta N}\right)^2 \sum |W||W'| \leq \frac{|\mathcal{P}|^2}{\delta^2 N^2}\delta' N^2 = \frac{1}{|\mathcal{P}|^2}.$$

Először felhasználtuk, hogy $\forall i \in [q] : |Q_i| \geq \frac{\delta N}{|\mathcal{P}|}$; majd azt, hogy a $\mathcal{W}$ partíció δ'-homogén; végül pedig azt, hogy $\delta' = \frac{\delta^2}{|\mathcal{P}|^4}$. Ismét a Boole-egyenlőtlenséget használva látható, hogy annak a valószínűsége, hogy minden (W_i, W_j) pár homogén, legalább

$$1 - \binom{q}{2}\frac{1}{|\mathcal{P}|^2} \geq 1 - \binom{|\mathcal{P}|}{2}\frac{1}{|\mathcal{P}|^2} \geq 1 - \frac{|\mathcal{P}|^2}{2}\frac{1}{|\mathcal{P}|^2} = 1/2.$$

Azt kaptuk, hogy $\Pr\left[\forall i \in [q] : |W_i| \geq \frac{N\delta^{4k^2-3k}}{2k^{4k}}\right] > 1/2$, valamit azt, hogy $\Pr[\forall 1 \leq i < j \leq q : (W_i, W_j) \text{ homogén}] \geq 1/2$. Tehát pozitív valószínűséggel mindkét esemény teljesül, vagyis a W_i halmazok kielégítik a lemma állítását. □

Egy $X \subseteq V(G)$ csúcshalmaz *sűrűség*e alatt a $d(X) = e(X)/\binom{|X|}{2}$ számot értjük, ahol $e(X)$ azoknak a G-beli éleknek a számát jelöli, amelyeknek mindkét végpontja X-ben van ($e(X) = |\{\{u, v\} \in E(G) : u, v \in X\}|$).

5.5. Lemma ([GS18] Lemma 10)**.** *Legyen G egy N csúcsú gráf, amelynek legalább αN^2 éle van, ahol $\alpha \in [0, 1/2)$. Ekkor tetszőleges $\beta \in (0, 1)$ esetén teljesül, hogy vagy van G-ben $\Omega(\alpha^{80}\beta^{20}N^4)$ darab feszített C_4 példány, vagy van olyan $X \subseteq V(G)$ csúcshalmaz, amelyre $|X| \geq 0{,}1\alpha^2 N$ és $d(X) \geq 1 - \beta$.*

Azt mondjuk, hogy (X, Y) ϵ-távol van a feszített-M_2-mentességtől, ha legalább $\epsilon|X||Y|$ élet kell módosítani (törölni vagy behúzni) X és Y között ahhoz, hogy (X, Y) feszített-M_2-mentessé váljon. Az 5.7. lemmához nem szerepel bizonyítás a cikkben, hanem egy általánosabb eredményre hivatkoznak [AFN07]. Ahhoz, hogy valamit megtudjunk az állításban szereplő konstansról, egy szorosan kapcsolódó eredményt fogalmazunk át bizonyítás címszó alatt.

5.6. Tétel ([Ver19] Theorem 6)**.** *Legyen $(X,Y)_G$ egy $|X|+|Y|=n$ csúcsú páros gráf. Ha legalább ϵn^2 élet kell módosítani ahhoz, hogy (X,Y)-t feszített-M_2-mentessé tegyük, akkor az egyenletes eloszlásból véletlenszerűen választott $\frac{4}{\epsilon}\ln\frac{1}{\epsilon}$ méretű $S\subseteq V(G)$ halmaz által feszített $(S\cap X, S\cap Y)_G$ gráf legalább 1/2 valószínűséggel tartalmaz feszített M_2 részgráfot.*

A következő lemma bizonyítása tehát nem része a forrásként használt cikknek, a benne említett d konstansról való ismeretszerzés céljából egészítettük ki vele a fő eredmény bizonyításának folyamát.

5.7. Lemma ([GS18] Lemma 12)**.** *Legyen az (X,Y) pár olyan, hogy $|X|=c_Xn$, $|Y|=c_Yn$ valamely $c_X,c_Y\in(0,1)$, $c_X+c_Y=1$ abszolút konstansokra, ahol $n=|X|+|Y|$. Ha (X,Y) ϵ-távol van a feszített-M_2-mentességtől, akkor (X,Y) legalább $\epsilon^d|X|^2|Y|^2$ feszített M_2 példányt tartalmaz, ahol $d>0$ abszolút konstans.*

Bizonyítás. Az, hogy (X,Y) ϵ-távol van a feszített-M_2-mentességtől, azt jelenti, hogy legalább $\epsilon|X||Y|=\epsilon c_Xc_Yn^2$ élet kell módosítani ahhoz, hogy (X,Y)-t feszített-M_2-mentessé tegyük. Az 5.6. tételt alkalmazzuk $\epsilon'=c_Xc_Y\epsilon$ paraméterrel. Eszerint az egyenletes eloszlásból véletlenszerűen választott $\frac{4}{\epsilon'}\ln\frac{1}{\epsilon'}$ méretű $S\subseteq V(G)$ halmaz által feszített $(S\cap X,S\cap Y)_G$ gráf legalább $1/2$ valószínűséggel tartalmaz feszített M_2 részgráfot.

Jelöljük m_2-vel az (X,Y)-ban található feszített M_2 példányok számát. Az S mintával $s=\binom{|S\cap X|}{2}\binom{|S\cap Y|}{2}\le\binom{|S|/2}{2}^2<\left(\frac{4}{\epsilon'}\ln\frac{1}{\epsilon'}\right)^4/64$ darab csúcsnégyest nézünk. Annak a valószínűsége, hogy egyetlen M_2-t sem találunk, $1/2>\left(1-\frac{m_2}{\binom{|X|}{2}\binom{|Y|}{2}}\right)^s$, amiből

$$m_2>\binom{|X|}{2}\binom{|Y|}{2}\left(1-2^{-1/s}\right)>\binom{|X|}{2}\binom{|Y|}{2}\left(1-2^{-64/\left(\frac{4}{\epsilon'}\ln\frac{1}{\epsilon'}\right)^4}\right)>$$
$$\binom{|X|}{2}\binom{|Y|}{2}\epsilon'^{19/3}=(c_Xc_Y)^{19/3}\binom{|X|}{2}\binom{|Y|}{2}\epsilon^{19/3}\approx\frac{(c_Xc_Y)^{19/3}}{4}|X|^2|Y|^2\epsilon^{19/3}.$$

A második egyenlőtlenségnél s felső becslésének behelyettesítése történt. A harmadik egyenlőtlenség helyessége számításokkal ellenőrizhető.

Ha $\epsilon\le\min\{c_X,c_Y\}\le1/2$, akkor m_2 értéke tovább becsülhető alulról $\epsilon^{21}|X|^2|Y|^2$-tel, azaz $d=21$ megfelelő. (De ha a konstans együtthatót figyelmen kívül hagyjuk, ami kis ϵ esetén teljesen indokolt, akkor látható, hogy valójában $d=19/3$ közelebb járhat a valósághoz.) □

5.8. Lemma ([GS18] Lemma 13)**.** *Létezik $c>0$ abszolút konstans, amelyre bármely $\alpha,\gamma\in(0,1)$ és N csúcsú G gráf esetén teljesül, hogy vagy van G-ben $\Omega(\alpha^c\gamma^cN^4)$ darab feszített C_4 példány, vagy van a csúcshalmazának olyan $V(G)=X_1\cup\cdots\cup X_k\cup Y$ partíciója, ahol $k\le10\alpha^{-5/2}$, és ami kielégíti az alábbi három feltételt:*

1. *$e(Y)<\alpha N^2$;*
2. *$\forall i\in[k]:|X_i|\ge0{,}1\alpha^{5/2}N$ és $d(X_i)\ge1-\gamma$;*
3. *Minden $1\le i<j\le k$ esetén (X_i,X_j) γ-közel van a feszített-M_2-mentességhez.*

5.8.1. Megjegyzés. *Az eredeti cikkben $|X_i|\ge0{,}1\alpha^3N$, $k\le10\alpha^{-3}$, és $c=\max\{84,20d\}$, míg itt $c=\max\{82,20d\}$. Mindhárom eltérés abból származik, hogy egy halmaz méretére pontosabb alsó becslést adtunk. (A cikkben a $|V_i|\ge\alpha N$ triviális becslést használják; mi kicsit több számolással a $|V_i|\ge\sqrt{\alpha}N$ becslést kaptuk.)*

Bizonyítás. A bizonyításban az abszolút konstans értéke $c=\max\{82,20d\}$ lesz, ahol d az 5.7. lemma konstansa. Definiáljuk a $\{V_i\}_{i\ge0}$ és az $\{X_i\}_{i\ge1}$ csúcshalmaz-sorozatokat

a következőképpen. Legyen $V_0 = V(G)$. Ha az i-edik lépésben $e(V_i) < \alpha N^2$, akkor itt megállunk. Vegyük észre, hogy az ellenkező esetben egyrészt $e(V_i) \geq \alpha N^2$ és $\binom{|V_i|}{2} \geq e(V_i)$ miatt $\binom{|V_i|}{2} \geq \alpha N^2$, amiből $|V_i|^2 - |V_i| \geq 2\alpha N^2$ és végül $|V_i| \geq 1/2 + \sqrt{1/4 + 2\alpha N^2} \geq \sqrt{\alpha} N$. Másrészt, ha $e(V_i) \geq \alpha N^2$, akkor alkalmazhatjuk az 5.5. lemmát a $G[V_i]$ gráfra α és $\beta = \gamma^d/4$ paraméterekkel. Ebből azt kapjuk, hogy vagy van $G[V_i]$-ben $\Omega(\alpha^{80}\gamma^{20d}|V_i|^4) \subseteq \Omega(\alpha^{82}\gamma^{20d}N^4)$ darab feszített C_4 példány – ekkor beláttuk a lemma állítását –, vagy van $X_{i+1} \subseteq V_i$, amelyre $|X_{i+1}| \geq 0{,}1\alpha^2|V_i| \geq 0{,}1\alpha^{5/2}N$ és $d(X_{i+1}) \geq 1 - \gamma^d/4$. Ez utóbbi esetben legyen $V_{i+1} = V_i \setminus X_{i+1}$, és ezzel kezdjük el a következő iterációt. Jelöljük k-val az utolsó megkezdett iteráció sorszámát: vegyük észre, hogy mivel az X_i halmazok diszjunktak és $V(G)$ részhalmazai, valamint $\forall i \in [k] : |X_i| \geq 0{,}1\alpha^2|V_i| \geq 0{,}1\alpha^{5/2}N$, a méreteik összegére $\sum_{i=1}^{k} |X_i| \leq N$, amiből következik, hogy $k \leq 10\alpha^{-5/2}$. Mivel a k-adik lépésben végeztünk, $e(V_k) < \alpha N^2$, és legyen $Y = V_k$. Így a $V(G)$ csúcshalmaznak egy olyan $\{X_1, \ldots, X_k, Y\}$ partícióját kaptuk, amelyre minden $i \in [k]$ esetén $|X_i| \geq 0{,}1\alpha^2|V_i| \geq 0{,}1\alpha^{5/2}N$ és $d(X_i) \geq 1 - \gamma^d/4 \geq 1 - \gamma$, valamint $e(Y) < \alpha N^2$. Az első két pontot tehát beláttuk.

A harmadik pontot indirekt módon bizonyítjuk: azt látjuk be, hogy ha a harmadik pont nem teljesül, akkor G-ben van legalább $0{,}5\gamma^d 10^{-4}\alpha^{10}N^4 \in \Omega(\alpha^c\gamma^c N^4)$ darab feszített C_4 példány. Tegyük fel, hogy van olyan i és j ($1 \leq i < j \leq k$), amelyre (X_i, X_j) γ-távol van a feszített-M_2-mentességtől. Az 5.7. lemmát használva (megtehetjük, mert $0{,}1\alpha^2|V_i| \leq |X_i| \leq |V_i|$ miatt a méretarányaik rendben vannak) tudjuk, hogy ekkor (X_i, X_j) legalább $\gamma^d|X_i|^2|X_j|^2$ feszített M_2 példányt tartalmaz. Ha egy ilyen példány két X_i-beli csúcsa között is van él, és a két X_j-beli csúcsa között is, akkor épp egy feszített C_4 példányt alkot a négy csúcs (lásd az 5.1. ábrát). Tudjuk, hogy $d(X_i)$ és $d(X_j)$ is legalább $1 - \gamma^d/4$, vagyis legfeljebb a csúcspárok $\gamma^d/4$ része között nincs él mindkét halmazon belül. Ebből következik, hogy legfeljebb $\gamma^d/2$ része olyan az $X_i \times X_i \times X_j \times X_j$-beli csúcsnégyeseknek, hogy akár az X_i-beli, akár az X_j-beli csúcspárja között nem megy él. Tehát a $\gamma^d|X_i|^2|X_j|^2$ darab feszített M_2 példány közül legfeljebb $|X_i|^2|X_j|^2\gamma^d/2$ darab nem határoz meg feszített C_4-et, azaz találtunk legalább $|X_i|^2|X_j|^2\gamma^d/2 \geq 10^{-4}\alpha^{10}N^4\gamma^d/2$ darab feszített C_4 példányt. □

5.9. Lemma ([GS18] Lemma 14)**.** *Létezik $c > 0$ abszolút konstans, amelyre bármely $\alpha, \gamma \in (0,1)$ és N csúcsú G gráf esetén teljesül, hogy vagy van G-ben $\Omega(\alpha^c\gamma^c N^4)$ feszített C_4 példány, vagy van egy G' gráf a $V(G)$ csúcshalmazon; egy $V(G) = X_1 \cup \cdots \cup X_k \cup Y$ partíció; egy $Z \subseteq X$ részhalmaz (ahol $X = X_1 \cup \cdots \cup X_k$); az $X \setminus Z$ halmaznak egy, a $\{X_1 \setminus Z, \ldots, X_k \setminus Z\}$ partíciót finomító $\mathcal{Q} = \{Q_1, \ldots, Q_q\}$ partíciója; és $W_i \subseteq Q_i$ részhalmazok, amik kielégítik az alábbi öt feltételt:*

1. *$G'[Y]$ független halmaz, $k \leq 10\alpha^{-5/2}$, és minden $i \in [k]$-ra $G'[X_i \setminus Z]$ klikk;*
2. *$|Z| < \alpha N$, és minden $z \in Z$ izolált csúcs G'-ben;*
3. *G'-ben $\mathrm{NH}(\mathcal{Q}) \leq \alpha N^2$;*
4. *$\forall i \in [q] : |W_i| \geq \frac{|X|}{2}\frac{\alpha^{400\alpha^{-5}+70\alpha^{-5/2}}}{10^{40\alpha^{-5/2}}}$, és G'-ben minden $1 \leq i < j \leq q$ esetén a (W_i, W_j) pár homogén;*
5. *$|E(G')\triangle E(G)| < (2\alpha + \gamma/2)N^2$, és $|E(G'[X \setminus Z])\triangle E(G[X \setminus Z])| < \gamma N^2/2$.*

5.9.1. Megjegyzés. *Az eredeti cikkben $|W_i| \geq |X|(\alpha/20)^{4000\alpha^{-6}}$ becslés szerepel, és az ötödik pontban két helyen $\gamma/2$ helyett γ áll (valamint az 5.8.1. megjegyzésben már említett módon $k \leq 10\alpha^{-3}$). Ezek egyrészt az előző lemmák javított változatainak használatából, másrészt pontosabb becslésekből származnak. (Pl. nálunk $|E(G''[X])\triangle E(G[X])| < \gamma N^2/2$, míg az eredeti cikkben γN^2 a becslés vége.)*

Bizonyítás. A c konstans legyen ugyanaz, mint az 5.8. lemma esetében. Alkalmazzuk G-re az 5.8. lemmát az adott α és γ paraméterekkel. Ha az első eset teljesül, azaz van G-ben $\Omega(\alpha^c\gamma^c N^4)$ darab feszített C_4 példány, akkor készen vagyunk. A második esetben kapunk egy $V(G) = X_1 \cup \cdots \cup X_k \cup Y$ partíciót, ahol $k \leq 10\alpha^{-5/2}$. Legyen G'' az a gráf, amelyet úgy kapunk G-ből, hogy Y-t független halmazzá, az X_i-ket ($\forall i \in [k]$) pedig klikkekké tesszük, valamint minden $1 \leq i < j \leq k$-ra az (X_i, X_j) párt feszített-M_2-mentessé változtatjuk. Az 5.8. lemma első pontja miatt $|E(G''[Y]) \triangle E(G[Y])| < \alpha N^2$, a második és harmadik pontból pedig következik, hogy $|E(G''[X]) \triangle E(G[X])| < \gamma \sum_{i=1}^{k} \binom{|X_i|}{2} + \gamma \sum_{i<j} |X_i||X_j| < \gamma/2 \left(\sum_{i=1}^{k} |X_i|^2 + 2\sum_{i<j} |X_i||X_j| \right) = \gamma/2 \left(\sum_{i=1}^{k} |X_i| \right)^2 < \gamma N^2/2$.

Alkalmazzuk az 5.4. lemmát $G''[X]$-re $\delta = \alpha$ paraméterrel az $\{X_1, \ldots, X_k\}$ partíció szerint. Így kapunk egy $Z \subseteq X$ halmazt, aminek a méretére $|Z| < \alpha|X| \leq \alpha N$ teljesül; és olyan $W_i \subseteq Q_i$ részhalmazokat, ahol minden $i \in [q]$-ra

$$|W_i| \geq \frac{|X|\alpha^{4k^2-3k}}{2k^{4k}} \geq \frac{|X|\alpha^{400\alpha^{-5}-30\alpha^{-5/2}}}{2 \cdot (10\alpha^{-5/2})^{40\alpha^{-5/2}}} = \frac{|X|}{2} \frac{\alpha^{400\alpha^{-5}+70\alpha^{-5/2}}}{10^{40\alpha^{-5/2}}}.$$

Legyen G' az a gráf, amelyet úgy kapunk G''-ből, hogy minden $z \in Z$ csúcsot izolálttá teszünk (mást nem változtatunk). Így az állítás második pontja rendben van. Az első pont G'' definíciója miatt teljesül, a harmadik és negyedik pontok pedig az 5.4. lemma, valamint az előző bekezdés indoklása miatt. Az ötödik pont második fele is rendben van, hiszen $G'[X \setminus Z] = G''[X \setminus Z]$, és $G''[X \setminus Z]$-re már beláttuk ugyanezt. Az ötödik pont első felét nézzük tehát. Mivel Z csúcsainak izolációjához legfeljebb $|Z|N$ élet kellett törölni, $|E(G') \triangle E(G'')| \leq |Z|N < \alpha N^2$. Ebből $|E(G') \triangle E(G)| \leq |E(G') \triangle E(G'')| + |E(G'') \triangle E(G)| < \alpha N^2 + (\alpha N^2 + \gamma N^2/2) = (2\alpha + \gamma/2)N^2$. □

5.10. Lemma ([GS18] Lemma 15)**.** *Legyen adott a G gráf és a csúcsainak egy olyan $V(G) = X \cup Y$ partíciója, ahol Y független halmaz, $G[X]$ pedig feszített-C_4-mentes. Ha legalább $\epsilon|X||Y|$ élet kell módosítani X és Y között ahhoz, hogy G feszített-C_4-mentessé váljon, akkor G tartalmaz legalább $\epsilon^4|X|^2|Y|^2/2^8$ darab feszített C_4 példányt.*

Ezek után nézzük magának az 5.1. tételnek a bizonyítását.

Az 5.1. tétel bizonyítása. Legyen

$$\alpha = \epsilon^6/2^{11} \text{ és } \gamma = \frac{\alpha^{1600\alpha^{-5}+280\alpha^{-5/2}}}{16 \cdot 10^{160\alpha^{-5/2}}} \left(\epsilon - \frac{\epsilon^6}{2^9} \right)^4.$$

Vegyük észre, hogy $\alpha > \gamma$, valamint, hogy valamely c' konstansra $\gamma > 1/2^{\epsilon^{-c'}}$ (egyszerű számítások után azt kaphatjuk, hogy például $c' = 102$ megfelelő). Alkalmazzuk az 5.9. lemmát G-re az imént definiált α és γ paraméterekkel. A lemma állításának első esetében van G-ben $\Omega(\alpha^c\gamma^c N^4) \subseteq \Omega(N^4/2^{2c\epsilon^{-c'}})$ darab feszített C_4 példány, vagyis teljesül a tétel állítása.

A második esetben az 5.9. lemma biztosítja a megfelelő G', $X = X_1 \cup \cdots \cup X_k$, Y, Z, $\mathcal{Q} = \{Q_1, \ldots, Q_q\}$ és $\{W_i\}_{i=0}^{q}$ halmazok, illetve partíciók létezését. Legyen G'' az a gráf, amelyet úgy kapunk G'-ből, hogy minden $1 \leq i < j \leq q$ esetén ha a (W_i, W_j) pár között futó élek teljes (illetve üres) páros gráfot határoznak meg, akkor a (Q_i, Q_j) pár között behúzunk (illetve kitörlünk) minden élet. Vegyük észre, hogy mivel minden (W_i, W_j) pár homogén (5.9. lemma, negyedik pont), a két eset egyike biztosan fennáll, azaz G''-ben minden (Q_i, Q_j) pár közötti páros gráf vagy üres, vagy teljes. Az 5.9. lemma harmadik pontja biztosítja, hogy legfeljebb αN^2 él változik meg, amikor G'-ből megkapjuk G''-t. Ebből, valamint az 5.9. lemma ötödik pontjából levezethető a következő felső becslés G''

és G távolságára: $|E(G'')\triangle E(G)| \leq |E(G'')\triangle E(G')| + |E(G')\triangle E(G)| < (3\alpha+\gamma/2)N^2 < 4\alpha N^2 = \frac{\epsilon^6}{2^9}N^2$. Mivel G a tétel állításának feltétele szerint ϵ-távol van a feszített-C_4-mentességtől, ez azt jelenti, hogy G'' távolsága ugyanettől a tulajdonságtól nagyobb, mint $\epsilon - \frac{\epsilon^6}{2^9}$. Ha $|X \setminus Z| < \left(\epsilon - \frac{\epsilon^6}{2^9}\right) N$ teljesülne, akkor minden $X \setminus Z$-re illeszkedő – legfeljebb $|X \setminus Z|N < \left(\epsilon - \frac{\epsilon^6}{2^9}\right) N^2$ darab – élet törölve az egész G''-t üres gráffá, és így feszített-C_4-mentessé tehetnénk. Ez ellentmond annak, hogy G'' legalább $\left(\epsilon - \frac{\epsilon^6}{2^9}\right)$-távol van a feszített-$C_4$-mentességtől, tehát azt kaptuk, hogy $|X \setminus Z| \geq \left(\epsilon - \frac{\epsilon^6}{2^9}\right) N$.

Két esetre bontjuk a további vizsgálódást: először tegyük fel, hogy a $G''[X \setminus Z]$ gráf tartalmaz egy feszített C_4 példányt. Legyenek $Q_{i_1}, Q_{i_2}, Q_{i_3}, Q_{i_4}$ azok a $\mathcal{Q}$-beli halmazok, amelyekben a C_4 példány egyes csúcsai benne vannak. Mivel G''-ben két $\mathcal{Q}$-beli halmaz között vagy minden él fut, vagy egyik sem, G''-ben minden $Q_{i_1} \times Q_{i_2} \times Q_{i_3} \times Q_{i_4}$-beli csúcsnégyes feszített C_4 példányt alkot. Azt is tudjuk, hogy G'-ben ugyanígy futnak az élek a megfelelő W_i részhalmazok között, tehát bármely $W_{i_1} \times W_{i_2} \times W_{i_3} \times W_{i_4}$-beli csúcsnégyes feszített C_4 példányt alkot G'-ben. Az 5.9. lemma negyedik pontja szerint tehát G' tartalmaz legalább

$$\prod_{j=1}^{4} |W_{i_j}| \geq \frac{|X|^4}{16} \frac{\alpha^{4(400\alpha^{-5}+70\alpha^{-5/2})}}{10^{4(40\alpha^{-5/2})}} \geq \frac{\left(\epsilon - \frac{\epsilon^6}{2^9}\right)^4 N^4}{16} \frac{\alpha^{1600\alpha^{-5}+280\alpha^{-5/2}}}{10^{160\alpha^{-5/2}}} = N^4\gamma$$

feszített C_4 példányt. Az ötödik pont szerint pedig tudjuk, hogy $G[X \setminus Z]$ és $G'[X \setminus Z]$ kevesebb mint $\gamma N^2/2$ élen különbözik. Mivel minden élet (csúcspárt) kevesebb mint N^2-féleképp lehet másik két csúccsal csúcsnégyessé kiegészíteni, minden él N^2-nél kevesebb feszített C_4 példányban vesz részt. Így $\gamma N^2/2$ él eltérésből kevesebb mint $\gamma N^4/2$ eltérés származhat a feszített C_4 példányok számában. Tehát a G'-beli γN^4 példány közül több mint $\gamma N^4/2$ megvan G-ben is, amivel beláttuk az állítást.

A másik eset, amikor a $G''[X \setminus Z]$ gráf feszített-C_4-mentes. Ekkor $G''[X]$ is az, mivel Z-ben csak izolált csúcsok vannak. Vegyük észre, hogy mivel $|X| + |Y| \leq N$, ezért $|X||Y| \leq N^2/4$, amiből következik, hogy $\left(\epsilon - \frac{\epsilon^6}{2^9}\right) N^2 \geq 4|X||Y| \left(\epsilon - \frac{\epsilon^6}{2^9}\right)$. Vezessük be az $\epsilon' = 4\left(\epsilon - \frac{\epsilon^6}{2^9}\right)$ paramétert. Tudjuk, hogy a G'' gráf $\left(\epsilon - \frac{\epsilon^6}{2^9}\right)$-távol van a feszített-$C_4$-mentességtől, ami azt jelenti, hogy nem tehető feszített-C_4-mentessé kevesebb mint $\left(\epsilon - \frac{\epsilon^6}{2^9}\right) N^2 \geq |X||Y|\epsilon'$ él módosításával X és Y között, ezért $|X||Y| \geq \left(\epsilon - \frac{\epsilon^6}{2^9}\right) N^2$. Alkalmazzuk az 5.10. lemmát G''-re ϵ' paraméterrel (megtehetjük, mert $G''[Y] = G'[Y]$ független halmaz az 5.9. lemma első pontja szerint; $G''[X]$ pedig feszített-C_4-mentes a feltételezésünk szerint). Azt kapjuk, hogy G'' tartalmaz legalább

$$\frac{\epsilon'^4}{2^8}|X|^2|Y|^2 = \frac{4^4\left(\epsilon - \frac{\epsilon^6}{2^9}\right)^4}{2^8}|X|^2|Y|^2 = \left(\epsilon - \frac{\epsilon^6}{2^9}\right)^4 |X|^2|Y|^2 \geq \left(\epsilon - \frac{\epsilon^6}{2^9}\right)^6 N^4$$

feszített C_4 példányt. Mivel $|E(G'')\triangle E(G)| < 4\alpha N^2$, és minden élhez legfeljebb N^2 darab C_4 példány tartozhat, G-ben is van legalább $\left(\epsilon - \frac{\epsilon^6}{2^9}\right)^6 N^4 - 4\alpha N^4 = \left(\left(\epsilon - \frac{\epsilon^6}{2^9}\right)^6 - \frac{\epsilon^6}{2^9}\right) N^4 \geq \left(\epsilon^6\left(1 - \frac{1}{2^9}\right)^6 - \frac{\epsilon^6}{2^9}\right) N^4 \geq 0{,}98\epsilon^6 N^4$ feszített C_4 példány. Ez jóval több, mint az elvárt mennyiség, vagyis végeztünk a bizonyítással. □

Nézzük meg, hogy összességében mi a különbség a [GS18]-beli eredmény és a jelenlegi között. Az eredeti cikkben is három ág van a fő tétel bizonyításában, az első-

ből $\Omega(\alpha^c\gamma^c N^4) \subseteq \Omega(\epsilon^{6c}2^{-c\epsilon^{-c'}}N^4)$, a másodikból $\gamma N^4 > 2^{-\epsilon^{-c'}}N^4$, a harmadikból pedig $N^4\epsilon^6/2^9$ darab feszített C_4 példány jön ki. Tudjuk azt is, hogy $c = \max\{84, 20d\}$, de sem d, sem c' értékére nem adnak becslést. Az 5.7. lemma bizonyítása alapján $d = 21$ megfelelő (bár az eredeti cikkben egy általánosabb eredményre hivatkoznak, amiből valószínűleg nagyobb konstans származna). Az 5.1. tétel bizonyításában említettünk egy módszert egy megfelelő c' kiszámítására: ezt az eredeti cikk γ-jára alkalmazva $c' = 122$-t kapunk. Így tehát a legrosszabb esetben $\Omega(\epsilon^{2520}2^{-420\epsilon^{-122}}N^4)$ a C_4-ek száma.

Nálunk a három eset közül az első kettőben a C_4 példányok száma első ránézésre megegyezik az előzőekkel: $\Omega(\alpha^c\gamma^c N^4) \subseteq \Omega(\epsilon^{6c}2^{-c\epsilon^{-c'}}N^4)$ az első, és $\gamma N^4 > 2^{-\epsilon^{-c'}}N^4$ a második becslés. A konstansok azonban már eltérnek: $c = \max\{82, 20d\}$, $d = 21$, valamint $c' = 102$. A harmadik esetben viszont jelentős javulást láthatunk: $0{,}98\epsilon^6N^4$-t kaptunk. A legrosszabb esetben $\Omega(\epsilon^{2520}2^{-420\epsilon^{-102}}N^4)$ a feszített C_4 példányok száma. Itt a γ érték leszorításával tudtunk valamit nyerni (ami a lemmák pontosabb változatainak használatával vált lehetővé), de továbbra is irreálisan kicsi számot ad a becslés. A legtöbben úgy sejtik, hogy a valóságban ϵ-ban polinomiális a C_4 példányok száma, tehát valószínűleg bőven van még tér további javításra. Ugyanakkor az is látható, hogy a bemutatott eredmény sokkal jobb, mint az eredeti, Szemerédi-lemmából levezetett változat, amely (pontosabban annak még az első javítása is) $1/\epsilon$-ban polinomiális magasságú kettőhatvány-torony reciprokával becsülte alulról a C_4-ek számát.

6. fejezet

Kapcsolódó kvantuminformatikai eredmények

A fejezetben először a tulajdonságtesztelés kvantuminformatikai megfelelőjét mutatjuk be néhány gyakran használt, alapvető szubrutinnal együtt. Ezután egy, a junták tesztelésére szolgáló kvantumalgoritmust vizsgálunk, amelyet összevetünk a 3.3. szakaszban tárgyalt klasszikus tesztelő algoritmussal. Végül egy lazán kapcsolódó témáról írunk, amely inkább az interaktív protokollok területéhez tartozik, és a CHSH-játék nevet viseli. Az első két szakasz részben Montanaro és de Wolf művén alapul [MW16].

6.1. Kvantum-tulajdonságtesztelés

Feltételezzük, hogy az olvasó rendelkezik alapvető kvantuminformatikai ismeretekkel. A következő szcenárióval dolgozunk: a vizsgált objektum egy (klasszikus) x karaktersorozattal adott, azaz bináris ábécét feltételezve $x \in \{0,1\}^N$. Ennek az i-edik bitjét egy $i \in [N]$ index esetén x_i-vel jelöljük. Ehhez az objektumhoz van orákulum-hozzáférése a kvantumalgoritmusnak, és most is az a fontos, hogy minél kevesebb kérdéssel oldjuk meg a problémákat.

De mit jelent kvantumos esetben egy lekérdezés? Természetesen egy unitér transzformációt, mégpedig az alábbi működéssel.

$$O_x : |i\rangle\,|\varphi\rangle \mapsto |i\rangle\,|\varphi + x_i\rangle$$

Tehát ha az első, $\log_2 N$ bites regiszterben megadjuk az i indexet, akkor a második, egybites regiszterhez hozzáadja x_i-t (mod 2). Ez akkor válik igazán érdekessé, ha az első regiszterben sok index szuperpozíciója van, és így egyetlen lekérdezéssel x sok bitjéről tudhatunk meg valamit.

Hasznos speciális eset, ha a második regiszterbe $|\varphi\rangle = |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ kerül. Ekkor az O_x lekérdezés hatása

$$O_x : |i\rangle\,|-\rangle \mapsto |i\rangle\,\frac{1}{\sqrt{2}}(|0+x_i\rangle - |1+x_i\rangle) = (-1)^{x_i}\,|i\rangle\,|-\rangle\,. \tag{6.1}$$

Ugyanis ha $x_i = 0$, akkor a második regiszter nem változik, marad $|-\rangle$; ha pedig $x_i = 1$, akkor megváltozik az előjele. Tehát azt látjuk, hogy a lekérdezett x_i érték fázisként jelent meg a végállapotban: ha $x_i = 0$, akkor pozitív az előjel (fázis), ellenkező esetben pedig negatív.

Egy kvantum tesztelő algoritmus során végzünk néhány tetszőleges (unitér) transzformációt, és közben néha egy-egy lekérdező transzformációt. Így T darab lekérdezés esetén

egy $\left|0^k\right\rangle$ kezdőállapotból induló általános algoritmus végállapota így írható:

$$|\psi_x\rangle = U_T O_x U_{T-1} O_x \dots O_x U_1 O_x U_0 \left|0^k\right\rangle .$$

A $|\psi_x\rangle$ végállapot az O_x lekérdező transzformációkon keresztül függ x-től. Végül ennek a végállapotnak a mérésével kapjuk meg a klasszikus végeredményt.

6.1.1. Amplitúdó-erősítés

Az amplitúdó-erősítés legismertebb megjelenése Grover kereső algoritmusában van [Gro96]. Ezt egy általánosan használható módszerként mutatjuk be. Hasonló elvről van szó, mint a POT-ok esetén. Tegyük fel, hogy van egy A kvantumalgoritmusunk, valamint orákulum-hozzáférésünk egy $\varrho : S \to \{0,1\}$ kiértékelő függvényhez. Azt tudjuk A-ról, hogy a kimenete egy olyan $w \in S$ szó, amelyre p valószínűséggel igaz, hogy $\varrho(w) = 0$. Ez megfeleltethető egy POT-nak: A biztosít valamilyen véletlen w objektumot, ami p eséllyel elutasításra kerül – ez felel meg annak, hogy $\varrho(w) = 0$. Ezt többször végrehajtva szeretnénk elérni, hogy 2/3 valószínűséggel utasítsunk el. Brassard és társai [BHMT02] eredménye alapján ehhez bármely $p > 0$ esetén elég $O(1/\sqrt{p})$ ismétlés. Tehát ha az A algoritmus ahhoz, hogy p eséllyel adjon jó w-t, q kérdést intézett ϱ-hoz, akkor összesen $O(q/\sqrt{p})$ kérdés elég a 2/3-os valószínűséghez.

Például a linearitás teszt (3.1. szakasz) esetén a POT 3 kérdést intéz ϱ-hoz, és a 3.2. tétel szerint (nem túl nagy ϵ esetén) legalább $\epsilon/2$ valószínűséggel utasít el, ha a függvény ϵ-távol van a linearitástól. Amplitúdó-erősítéssel olyan linearitástesztelő kvantumalgoritmust kapunk, aminek a lekérdezésbonyolultsága $O(3/\sqrt{\epsilon/2}) = O(1/\sqrt{\epsilon})$. Ehhez képest ugyanez az érték klasszikus esetben $O(1/\epsilon)$ volt.

6.1.2. Bernstein–Vazirani-algoritmus

A Bernstein–Vazirani-algoritmus egy korai kvantumalgoritmus [BV97]. Orákulum-hozzáféréssel adott egy $f : \{0,1\}^n \to \{0,1\}$ függvény, amelyről tudjuk (ígéretproblémaszerűen), hogy valamely $s \in \{0,1\}^n$ vektorral vett skalárszorzatot számol, vagyis $f(x) = x \cdot s = \sum_{i\in[n]} x_i s_i \pmod 2$. Észrevehetjük, hogy ez egyfajta linearitásnak felel meg. A feladat minél kevesebb lekérdezéssel meghatározni s értékét. Klasszikusan a leghatékonyabb megoldás minden $x = 2^k$-ra ($k \in \{0,1,\dots,n-1\}$) lekérdezni a függvény értékét, és így s egy-egy bitjét megtudni: ez n kérdés. Kvantumosan egyetlen lekérdezés elegendő.

Induljunk ki az n qubites csupa 0 állapotból, kiegészítve egy 1-essel. Erre alkalmazzuk az $n+1$ bites Hadamard-transzformációt, amitől az első n biten egyenletes szuperpozíciót kapunk, $|1\rangle$-ből pedig $|-\rangle$-t. A kapott állapoton alkalmazzuk az O_f lekérdező transzformációt, ami (6.1) alapján a fázisban jeleníti meg $f(x)$-et. Még egy Hadamard-transzformációt végzünk, amiről tudjuk, hogy $(H_{n+1})_{i,j} = (-1)^{i\cdot j}/\sqrt{2^{n+1}}$. Ezután olyan állapotot kapunk, amiről tudjuk, hogy az első n bitje 1 valószínűséggel $|s\rangle$.

$$|0^n\rangle|1\rangle \xrightarrow{H_{n+1}} \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle|-\rangle \xrightarrow{O_f} \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle|-\rangle \xrightarrow{H_{n+1}}$$
$$\frac{1}{2^n}\sum_{x=0}^{2^n-1}(-1)^{f(x)}\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}|y\rangle|1\rangle = \frac{1}{2^n}\sum_{x,y=0}^{2^n-1}(-1)^{x\cdot s+x\cdot y}|y\rangle|1\rangle = |s\rangle|1\rangle$$

Annak megértéséhez, hogy miért kapjuk meg végül $|s\rangle$-t, nézzük meg, hogy a végállapotban mi egy tetszőleges, rögzített y valószínűségi amplitúdója: $\frac{1}{2^n}\sum_{x=0}^{2^n-1}(-1)^{x\cdot(s+y)}$, ahol $s+y$ bitenkénti összeadást jelent. Ha $y = s$, akkor a -1 kitevője minden x-re 0,

azaz az amplitúdó 1. Bármely egyéb y esetén az x-ek fele ad 0, a másik fele 1 kitevőt, így ugyanannyi $+1$ és -1 adódik össze, vagyis az amplitúdó 0. Tehát az $y = s$ eset teljesül 1 valószínűséggel. Így a végállapot első n qubitjét a standard bázisban megmérve 1 valószínűséggel megkapjuk s-t.

Megjegyezzük, hogy az algoritmus $f : \{0,1\}^n \to \{0,1\}^{2^n}$ függvényekre is működik, és ebben az esetben például Hadamard-kódszavak részhalmazaiba való tartozás tesztelésére használható. Az így kapott kvantumalgoritmus hatékonyabb a legjobb klasszikus algoritmusnál.

6.2. Junták tesztelése

A 3.3. szakaszban írtaknak megfelelően az $f : \{0,1\}^n \to \{0,1\}$ függvény k-junta, ha az értéke legfeljebb k változótól függ. Ahogy azt már említettük, klasszikusan a legjobb ismert ϵ-tesztelő algoritmus $O(k \log k + k/\epsilon)$ kérdést tesz fel, $\Omega(k)$ pedig alsó korlát a lekérdezésbonyolultságra. Atıcı és Servedio [AS07] $O(k/\epsilon)$ kérdést használó kvantumalgoritmust adott a problémára, ami amplitúdó-erősítéssel tovább javítható $O(k/\sqrt{\epsilon})$-ra. Az algoritmus lényegében a Bernstein–Vazirani-algoritmust hajtja végre $O(k/\sqrt{\epsilon})$ alkalommal. A 6.1. algoritmus mutatja a működést; az $(n+1)$-edik kvantumbitet itt nem tüntettük fel.

Bemenet:
Orákulum-hozzáférés az $f : \{0,1\}^n \to \{0,1\}$ függvényhez;
k paraméter;
ϵ távolságparaméter.

A fő lépések:

1. Legyen $W_v = (00\ldots0)$ az n hosszú nullvektor.
2. **Ciklus** $O(k/\sqrt{\epsilon})$ körön át
   - 2.1. Alkalmazzunk Hadamard-transzformációt a kezdeti $|0^n\rangle$ állapotra, így kapjuk az egyenletes szuperpozíciót: $\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$.
   - 2.2. Az O_f lekérdező transzformáció hatására az új állapot $\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle$.
   - 2.3. Újabb Hadamard-transzformációval kapjuk a végső, $\frac{1}{2^n} \sum_{x,y=0}^{2^n-1} (-1)^{f(x)+x\cdot y} |y\rangle$ állapotot.
   - 2.4. Mérjük meg a végállapotot, és a kapott S_v vektort bitenkénti vagy művelettel adjuk hozzá W_v-hez: $W_v \leftarrow W_v + S_v$.
3. **Ha** W_v-ben több mint k darab egyes van, **akkor** utasítsunk el; **egyébként** fogadjunk el.

6.1. Algoritmus: Kvantumalgoritmus k-junta tesztelésre.

6.1. Tétel. *A 6.1. algoritmus ϵ-tesztelő algoritmus a k-junta tulajdonságra. A lekérdezésbonyolultsága $O(k/\sqrt{\epsilon})$.*

Bizonyítás. A lekérdezésbonyolultságra vonatkozó állítás nyilvánvaló, hiszen a Bernstein–Vazirani-algoritmus egyetlen kérdést tesz fel, és most ezt ismételjük $O(k/\sqrt{\epsilon})$-szor.

A további vizsgálathoz érdemes bevezetni az $f' : \{0,1\}^n \to \{-1,+1\}$ függvényt, amelyre $f'(x) = (-1)^{f(x)}$. Tehát f' ugyanúgy működik, mint f, csak 0 helyett $+1$-et, 1 helyett pedig -1-et vesz fel. Így a mérés előtti állapot így is írható: $\frac{1}{2^n} \sum_{x,y=0}^{2^n-1} f'(x)(-1)^{x\cdot y} |y\rangle$.

Vezessük be a következő Fourier-együtthatókat:

$$\forall y \in \{0,1\}^n \text{ esetén } \hat{f}(y) = \frac{1}{2^n} \sum_{x=0}^{2^n-1} f'(x)(-1)^{x \cdot y}. \tag{6.2}$$

Ezt felhasználva a mérés előtti állapot $\sum_{y=0}^{2^n-1} \hat{f}(y) \, |y\rangle$ alakba írható. Ez azt jelenti, hogy a mérés során minden $y \in \{0,1\}^n$-re $\hat{f}^2(y)$ a valószínűsége annak, hogy y-t kapjuk.

Legyen $J \subseteq [n]$ a „befolyásos" változók halmaza – tehát a kérdés az, hogy $|J| \leq k$ teljesül-e. Ezentúl egy $S \subseteq [n]$ halmaz esetén az $S_v \in \{0,1\}^n$ vektor az S által meghatározott karakterisztikus vektort jelöli. Vegyük észre, hogy ha $S \nsubseteq J$, akkor $\hat{f}(S_v)$ (6.2) képletében kinullázódik az összeg: a J-beli pozíciókon tetszőleges rögzített értékeket felvevő x vektorok felének 0, másik felének 1 a skalárszorzata S_v-vel, mégpedig S_v-nek a J-beli pozíciókon kívül eső egyesei miatt. Tehát $\hat{f}(S_v) \neq 0$-ból következik, hogy $S \subseteq J$, vagyis a mérés eredményeként csak befolyásos változókat tartalmazó halmaz karakterisztikus vektorát kaphatjuk. Ebből következik, hogy $W \subseteq J$ mindig teljesül, azaz ha f valóban k-junta, akkor a 6.1. algoritmus biztosan elfogad.

De mi a helyzet, ha f ϵ-távol van minden k-juntától? Legfeljebb k méretű $W \subseteq [n]$ esetén legyen $g_W : \{0,1\}^n \to [-1,1]$ a következő függvény: $g_W(x) = \sum_{S \subseteq W} \hat{f}(S_v)(-1)^{x \cdot S_v}$; valamint $h_W(x) = \operatorname{sgn}(g_W(x))$. Mivel $|W| \leq k$ és $S \subseteq W$, az S_v vektorban legfeljebb k darab egyes van. A h_W függvény pedig az x bemeneti vektortól csak annak az S_v-vel való skalárszorzatán keresztül függ, vagyis legfeljebb k darab x-beli változótól. Tehát h_W egy k-junta, amiből következik, hogy f ϵ-távol van tőle.

$$\epsilon \leq \Pr\big[f'(x) \neq h_W(x)\big] = \frac{1}{2^n} \sum_{x : f'(x) \neq h_W(x)} 1 \leq \frac{1}{2^n} \sum_{x : f'(x) \neq h_W(x)} (f'(x) - g_W(x))^2$$

Az utolsó lépés azért helyes, mert ha $f'(x) = 1$, akkor $f'(x) \neq h_W(x)$ miatt $h_W(x) = -1$ vagy 0. Ez pedig azt jelenti, hogy $g_W(x) \leq 0$, vagyis $(f'(x) - g_W(x))^2 \geq 1$. Hasonlóan, $f'(x) = -1$ esetén $h_W(x) = 1$ vagy 0, azaz $g_W(x) \geq 0$, amiből szintén következik, hogy $(f'(x) - g_W(x))^2 \geq 1$. Az előző becslés tovább folytatható az alábbi módon.

$$\epsilon \leq \frac{1}{2^n} \sum_{x \in \{0,1\}^n} (f'(x) - g_W(x))^2 = \sum_{S \subseteq [n]} (\hat{f}(S_v) - \hat{g_W}(S_v))^2 = \sum_{S \nsubseteq W} \hat{f}^2(S_v)$$

Az első egyenlőség a Parseval-azonosság, azaz $\frac{1}{N} \sum_{x=1}^{N} \varphi^2(x) = \sum_{y=1}^{N} \hat{\varphi}^2(y)$ miatt teljesül, hiszen az $(f'(x) - g_W(x))$ függvény Fourier-együtthatói (6.2) alapján az $(\hat{f}(S_v) - \hat{g_W}(S_v))$ értékek. Az utolsó egyenlőség indoklásához fejtsük ki $\hat{g_W}(S_v)$ értékét.

$$\hat{g_W}(S_v) = \frac{1}{2^n} \sum_{x=0}^{2^n-1} g_W(x)(-1)^{x \cdot S_v} = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{x \cdot S_v} \sum_{Z \subseteq W} \hat{f}(Z_v)(-1)^{x \cdot Z_v} =$$

$$\frac{1}{2^n} \sum_{Z \subseteq W} \sum_{x=0}^{2^n-1} (-1)^{x \cdot (S_v + Z_v)} \hat{f}(Z_v) = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{(x \cdot (S_v + S_v))} \hat{f}(S_v) = \hat{f}(S_v)$$

Az első három egyenlőség egyszerű behelyettesítések és átrendezések. Az utolsó előtti egyenlőség már csak akkor teljesül, ha $S \subseteq W$. Ugyanis azt használtuk, hogy ha egy tetszőleges rögzített $Z \neq S$ halmazt nézünk, akkor a minden x-re történő összegzés kinullázódik. Tehát csak a $Z = S$ eset marad meg – és ilyen eset csak akkor van, ha a W

részhalmazain végigfutó Z valamikor éppen S lesz, azaz ha $S \subseteq W$. Az utolsó egyenlőség azért igaz, mert $S_v + S_v = 0$, azaz 2^n darab 1-est adunk össze. Tehát azt kaptuk, hogy ha $S \subseteq W$, akkor $g_{\hat{W}}(S_v) = \hat{f}(S_v)$, egyébként pedig $g_{\hat{W}}(S_v) = 0$.

Végül is azt láttuk be, hogy $\epsilon \leq \sum_{S \not\subseteq W} \hat{f}^2(S_v)$, vagyis legalább ϵ valószínűséggel a mérés eredménye olyan halmazhoz tartozó karakterisztikus vektor, ami nem részhalmaza W-nek. Amplitúdó-erősítéssel azt kapjuk, hogy várhatóan $O(1/\sqrt{\epsilon})$ ismétléssel kapunk ilyen S_v-t, és ekkor végrehajtjuk a $W \leftarrow W \cup S$ lépést. Egy ilyen alkalommal legalább eggyel nő W mérete, vagyis mindezt $(k+1)$-szer megismételve $|W| > k$ teljesül, amikor is elutasítunk. □

Megjegyezzük, hogy alkottak már az itt bemutatottnál hatékonyabb – de bonyolultabb – kvantumalgoritmust is junták tesztelésére [ABRW16]. Ez $\tilde{O}(\sqrt{k/\epsilon})$ kérdést tesz fel, amivel az $\Omega(k)$ klasszikus alsó korlát alatt van. A szerzők azt is belátták a cikkben, hogy kvantumosan szükség van legalább $\Omega(k^{1/3})$ kérdésre.

6.3. CHSH-játék avagy Bell-egyenlőtlenség

A CHSH betűk Clauser, Horne, Shimony és Holt neveiből származnak. Az egyenlőtlenség jelentősége, hogy megmutatja, hogy a kvantum összefonódás következményei nem reprodukálhatók klasszikus esetben, rejtett változókat feltételezve.

Az 1.2. definíció szerinti $k = 2$ személyes játékot veszünk, a két játékos Alice és Bob. A játék a következőképpen írható le:

- $Q_1 = Q_2 = \{0, 1\}$;
- π egyenletes eloszlás (azaz mind a 4 eset valószínűsége 1/4);
- $A_1 = A_2 = \{0, 1\}$;
- $V(a, b, s, t) = [(a \text{ XOR } b) = (s \text{ AND } t)] = [a + b \equiv st \pmod 2]$.

Vagyis a bíró küld egy-egy bitnyi s, illetve t kérdést rendre Alice-nek és Bobnak, erre ők rendre a szintén egybites a-val, illetve b-vel válaszolnak. Akkor nyernek a játékosok, ha úgy válaszolnak, hogy a válaszaik XOR (kizáró vagy) kapcsolata ugyanaz, mint a bíró kérdéseinek AND (és) kapcsolata.

6.3.1. Klasszikus eset

Tekintsük az a és b válaszokat a kérdések függvényeinek (a korábbi jelöléssel rendre f_1 és f_2). A játék értéke (az 1.2. definíció alapján), bevezetve a $p_{a,b}$ jelölést

$$\omega = \max_{a,b} \sum_{s=0}^{1} \sum_{t=0}^{1} \frac{1}{4} V(a(s), b(t), s, t) = \max_{a,b} p_{a,b};$$

$$p_{a,b} = \Pr[\text{játékosok nyernek} | \text{stratégiájuk } a, \text{ illetve } b];$$

$$p_{a,b} = \sum_{s=0}^{1} \sum_{t=0}^{1} \frac{1}{4} V(a(s), b(t), s, t) = \frac{1}{4} \sum_{s=0}^{1} \sum_{t=0}^{1} [a(s) + b(t) \equiv st \pmod 2].$$

Nézzük azt a stratégiát, amikor Alice és Bob is bármilyen kérdésre 0-t válaszol, azaz $\forall s, t \in \{0, 1\}$ esetén $a(s) = b(t) = 0$. Ekkor $a(s) + b(t) = 0$ mindig igaz, az st pedig csak akkor 1, ha $s = t = 1$, a többi három esetben 0. Tehát ilyenkor $p_{a,b} = 3/4$. Az ω a $p_{a,b}$-k maximuma, amelyek értékkészlete $\{0,\ 1/4,\ 1/2,\ 3/4,\ 1\}$ (attól függően, hogy a

négy lehetőség közül hány esetben teljesül $a(s)+b(t) \equiv st \pmod 2$). Mivel találtunk olyan esetet, amire $p_{a,b} = 3/4$, az ω már csak 3/4 vagy 1 lehet.

Tegyük fel, hogy $\omega = 1$. Ekkor $a(s) + b(t) \equiv st \pmod 2$ mind a négy esetben igaz, azaz $a(s) + b(t) \equiv 1 \pmod 2$ csak $s = t = 1$ esetben teljesül (a többi 3 esetben 0). Ezért a négy esetet összegezve azt kapjuk, hogy $\sum_{s=0}^{1}\sum_{t=0}^{1} a(s) + b(t) \equiv 1 \pmod 2$. Másrészt $\sum_{s=0}^{1}\sum_{t=0}^{1} a(s) + b(t) = 2(\sum_{s=0}^{1} a(s) + \sum_{t=0}^{1} b(t)) \equiv 0 \pmod 2$, hiszen páros. Ezzel ellentmondásra jutottunk, így $\omega = 3/4$.

6.3.2. Kvantumos eset

A korábbiakhoz képest annyi a különbség, hogy Alice és Bob osztozik egy összefonódott $|\varphi\rangle$ kvantumbit-páron. Alice-nek van két projektív mérése (PVM): egy-egy a lehetséges $s \in \{0,1\}$ kérdésekre. Formálisan: $E_s = \{E_s(0), E_s(1)\}$, ahol minden $s, x \in \{0,1\}$ esetén $E_s(x)^2 = E_s(x)$, és minden s-re $E_s(0)+E_s(1) = I$. Bobnak hasonlóan: $G_t = \{G_t(0), G_t(1)\}$.

A stratégiát most a $|\varphi\rangle$, az E_s-ek és a G_t-k összessége jelenti. Ezek rögzítésével

$$\Pr[\text{nyerés}] = \sum_{x,y,s,t} V(x,y,s,t)\Pr[x,y|s,t]\frac{1}{4};$$

$$\Pr[x,y|s,t] = \langle\varphi| \left(E_s(x) \otimes G_t(y)\right) |\varphi\rangle .$$

A játék értékének definícióját módosítani kell, mert kvantumos esetben nem biztos, hogy létezik a maximum – így a szuprémumot vesszük helyette. Ettől eltekintve analóg a klasszikus esettel: a legjobb stratégiához tartozó nyerési valószínűség. A klasszikus esettől való megkülönböztetés miatt ezt kvantum értéknek nevezzük, és ω^*-gal jelöljük:

$$\omega^* = \sup_{|\varphi\rangle, E_s, G_t} \Pr[\text{nyerés}].$$

Azt fogjuk megmutatni, hogy van olyan stratégia, amelyre $\Pr[\text{nyerés}] > 3/4$, amiből következik, hogy $\omega^* > 3/4$, azaz kvantumosan van jobb nyerési stratégia, mint klasszikusan.

A stratégia megadása $|\varphi\rangle, E_s, G_t$ megadását jelenti. Legyen $|\varphi\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$. Definiáljuk a $|\alpha(\theta)\rangle = \frac{|0\rangle+e^{i\theta}|1\rangle}{\sqrt{2}}$ állapotokat (ezek a Bloch-gömb egyenlítőjén helyezkednek el). Ennek a segítségével legyen

$$E_s(x) = |\alpha(\delta_1)\rangle \langle\alpha(\delta_1)| \text{ ahol } \delta_1 = \pi \cdot \left(\frac{s}{2} + x\right);$$

$$G_t(y) = |\alpha(\delta_2)\rangle \langle\alpha(\delta_2)| \text{ ahol } \delta_2 = \pi \cdot \left(\frac{t}{2} + y - \frac{1}{4}\right).$$

Ezekre már kiszámítható a keresett érték:

$$\Pr[x,y|s,t] = \langle\varphi| \left(|\alpha(\delta_1)\rangle \langle\alpha(\delta_1)| \otimes |\alpha(\delta_2)\rangle \langle\alpha(\delta_2)|\right) |\varphi\rangle =$$

$$= \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 0 & 0 & 1\end{bmatrix}\left(\frac{1}{\sqrt{2}}\begin{bmatrix}1 & e^{-i\delta_1}\\ e^{i\delta_1} & 1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & e^{-i\delta_2}\\ e^{i\delta_2} & 1\end{bmatrix}\right)\frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\0\\1\end{bmatrix} =$$

$$= \frac{1}{4}\left(1 + \mathrm{Re}(e^{i(\delta_1+\delta_2)})\right) = \frac{1}{4}\left(1 + \mathrm{Re}(e^{i\pi(x+y+\frac{s+t}{2}-\frac{1}{4})})\right) =$$

$$= \frac{1}{4}\left(1 + (-1)^{x+y}\frac{1}{\sqrt{2}}(-1)^{st}\right) = \frac{1}{4}\left(1 + \frac{1}{\sqrt{2}}(-1)^{x+y+st \pmod 2}\right).$$

Ebből

$$\Pr[\text{nyerés}] = \frac{1}{4}\sum_{x,y,s,t} V(x,y,s,t)\Pr[x,y|s,t] =$$

$$= \frac{1}{4}\sum_{x,y,s,t} [x+y+st \equiv 0 \pmod 2]\frac{1}{4}\left(1+\frac{1}{\sqrt{2}}(-1)^{x+y+st \pmod 2}\right) =$$

$$= \frac{1}{2}\left(1+\frac{1}{\sqrt{2}}\right) \approx 0{,}85 > 3/4.$$

Ezzel beláttuk az állítást. Fizikai kísérletekben a bíró két átellenes oldalán van Alice és Bob, és fénysebességgel zajlik a kommunikáció. A bíró egyszerre küldi el a két kérdést, és arra a játékosok rögtön válaszolnak. Így kizárt, hogy Alice és Bob kommunikáljanak egymással.

7. fejezet

Elméleti eredmények verifikálása

A korábbi fejezetekben tárgyalt tulajdonságtesztelő algoritmusok közül a linearitás tesztet és a gráfok párosságának tesztelését valósítottunk meg programként. A programok C++ nyelven készültek a Microsoft Visual Studio 2019 fejlesztőkörnyezetben.

7.1. Linearitás teszt

A 3.1. alfejezettől eltérően itt f nem feltétlenül bináris, az általánosabb változatot teszteljük. A szoftver az elméleti eredményeknél már említett fekete doboz megadási mód mellett egy táblázatos változatot is támogat.

Táblázatos megadás esetén olyan bemeneti fájlt várunk, ahol soronként két érték szerepel szóközzel elválasztva: egy x és a hozzá tartozó $f(x)$. Az adatokat beolvasás után map-ben tároljuk. Mivel véges sok értéket tudunk eltárolni, a programban a természetes számok egy véges I részhalmazából valók lehetnek az x értékek. Ilyen megadáshoz készült egy adatgeneráló kód is, ami a kísérlethez használt adatok esetén a $[0; 999]$ intervallumbeli egészekhez rendelt egy-egy valós értéket. Szinte minden adat egy egyenesre illeszkedik, kivéve várhatóan az 1/1000 részük. (Ezekre $f(x) = ax + 1$, így ha pl. x és $x + y$ is ilyen, de y nem, akkor nem észleljük a nemlinearitást, de ezt elhanyagoljuk.)

A fekete doboz a programban egy olyan objektumot jelent, amelynek egy publikus tagfüggvénye a paraméterül kapott x bemenetre visszatér $f(x)$-szel. Az objektum a konstruktorában generál egy véletlen a számot, és a 3.1. tétel alapján általában az $f(x) = ax$ lineáris függvénynek megfelelő a válasza. Létrehozáskor megadható, hogy az esetek hányadrészében adjon mást.

Mindkét megadási esetben a 7.1. algoritmust futtatjuk. Különbség, hogy az egyik esetben az $f(x)$ értéket egy map-ből olvassuk ki, míg a másik esetben egy metódushívás történik. Ezenkívül (ha $f : G \to H$) a G az első esetben az I halmaz, míg a másodikban a valós számok azon részhalmaza, amelyek számítógéppel kezelhetők. H mindkét esetben a valósak halmaza. További különbség, hogy ha az első esetben olyan x értékhez kérünk $f(x)$-et, ami nincs eltárolva, akkor a belső ciklust a ciklusváltozó növelése nélkül újrafuttatjuk.

A 7.1. algoritmusban a POT-ot, azaz a 3.1. algoritmust k-szor futtatjuk, és feljegyezzük, hogy volt-e olyan eset, ami nemlinearitást jelzett. Ezt megismételjük T-szer, hogy megnézzük, az eseteknek hányadrészében lehet k ismétléssel kiszűrni egy, a linearitástól ϵ-távoli függvény nemlinearitását.

A kísérlet során mind a LUT (táblázatos), mind a BB (fekete doboz) megadás esetén olyan f-et modelleztünk, amelyre létezik a, hogy a legtöbb bemenetre $f(x) = ax$, azonban az esetek $\epsilon = 1/1000$ részében ettől eltérő a függvény értéke. A további paraméterek: $T = 10000$, k-t pedig 10-től tízesével növeltük 1000-ig. A kapott eredményeket a 7.1. ábrán látható diagram mutatja.

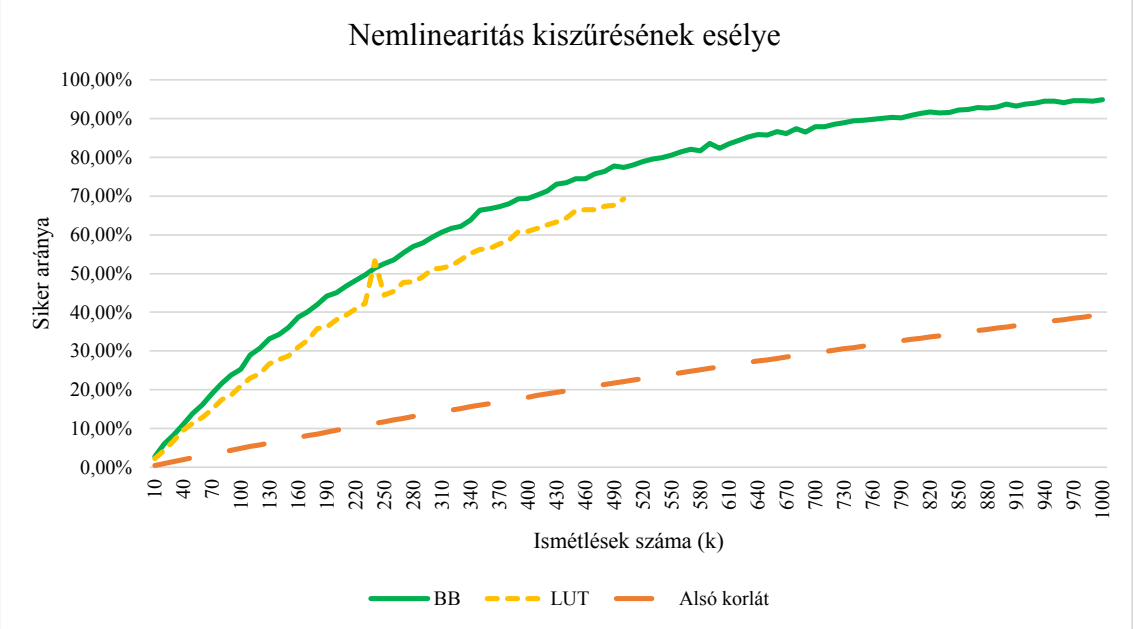
Bemenet:
Orákulum-hozzáférés az $f : G \to H$ függvényhez;
k és T paraméterek.

A fő lépések:

1. számláló $= 0$.
2. **Ciklus** T körön át
   2.1. **Ciklus** k körön át
      2.1.1. Hajtsuk végre a 3.1. algoritmust f-re.
      2.1.2. **Ha** elutasít, **akkor** növeljük eggyel a számláló értékét,
      és lépjünk ki a belső ciklusból.

Kimenet: számláló, azaz a T közül hány esetben találtunk nemlinearitást.

7.1. Algoritmus: A programban végrehajtott, linearitást tesztelő algoritmus.



7.1. ábra. A nemlinearitást észrevevő futtatások aránya különböző k értékek és egyváltozós függvény esetén.

A LUT változatot csak $k = 500$-ig futtattuk, mert lassabban futott, mint a BB. De már az addigi eredményekből is egyértelműen látszik, hogy a LUT valamivel rosszabbul teljesít. Ennek oka, hogy véges sok, pl. 1000 értéket tartalmaz a táblázat, és ha ezek közül csak egy van, ami nem illeszkedik a többi egyenesére, akkor ezt csak úgy lehet kiszűrni, ha őt kisorsoljuk egy nem túl nagy másik elemmel (hogy még az összegükhöz tartozó értéket is tartalmazza a táblázat) – vagy ha két kisebb elemet sorsolunk ki, amelyek összege az egyenesre nem illeszkedő elem. A nagyobb futásidőt az magyarázza, hogy ha olyan elemeket sorsolunk, amelyek összegéhez nincs érték rendelve a táblázatban, akkor újra kell sorsolni.

A 3.2. tétel állítását a kísérlet paramétereire alkalmazva azt kapjuk, hogy a 3.1. algoritmus $\frac{\epsilon}{2} = \frac{1}{2000}$-nél nagyobb valószínűséggel talál nemlinearitást. Ez alapján k független futtatás esetén legalább $1 - (1 - 1/2000)^k$ a nemlinearitás kiszűrésének esélye. Ez az adatok alapján teljesülni látszik, sőt ennél jelentősen nagyobb valószínűségeket látunk. Pl. $k = 500$ esetén az alsó becslés $\approx 0{,}22$, de a kísérlet során $\approx 0{,}77$ eredményt kaptunk. Te-

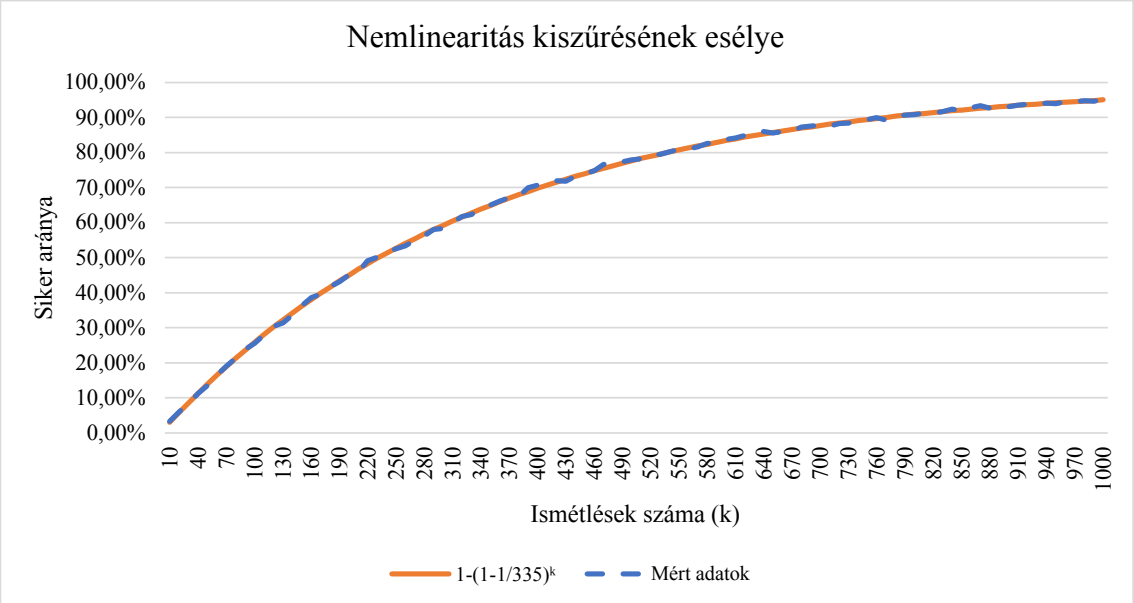
hát a gyakorlatban az elméleti eredménynél jóval kevesebb iterációval is elérhető a kívánt valószínűség.

Többváltozós eset

Az egyváltozós változat tapasztalatai alapján több változóra már csak a fekete doboz (BB) megadási módot valósítottuk meg. Egyébként a program az egyváltozós esethez nagyon hasonlóan működik, azonos algoritmust futtatunk.

A változók száma egy paraméter: ekkora méretű tömbökkel dolgozik a program. Az x és az y most többdimenziós vektorok (ezért a továbbiakban $\underline{x}$-szel, illetve $\underline{y}$-nal jelöljük őket), de a linearitást most is ugyanúgy ellenőrizzük: $f(\underline{x}) + f(\underline{y}) = f(\underline{x} + \underline{y})$ teljesül-e. Az egyes $f(\underline{x})$ értékek kiszámolása is hasonlóan történik, mint egyetlen változó esetén, de most nem egy véletlen a számmal, hanem egy ($\underline{x}$-szel azonos méretű) véletlen elemekből álló $\underline{a}$ vektorral szorozzuk össze skalárisan $\underline{x}$-et. Tehát általában $f(\underline{x}) = \underline{a} \cdot \underline{x}$, kivéve az esetek egy kis – pl. 1/1000 – részében, amikor pedig $f(\underline{x}) = \underline{a} \cdot \underline{x} + 1$.

Az egyváltozós esethez hasonlóan $T = 10000$ és $\epsilon = 1/1000$ értékekkel futtattuk a 7.1. algoritmust: $k = 10$-től 1000-ig változott, tízesével növelve. A változók száma 5 volt. A 7.2. ábrán láthatók az eredmények, vagyis az, hogy az egyes k értékek esetén a 10000 futtatás hány százalékában sikerült észrevennie az algoritmusnak, hogy a linearitástól ϵ-távoli függvény nem lineáris. Látható, hogy a grafikon lényegében egybeesik a 7.1. ábra zöld grafikonjával – azért nem ábrázoltuk mindkét adatsort egy diagramon, mert egészen fedik egymást –, azaz a változók száma nem befolyásolja az adott sikervalószínűséghez szükséges ismétlések számát.



7.2. ábra. A nemlinearitást észrevevő futtatások aránya különböző k értékek és ötváltozós függvény esetén.

Amikor arra a kérdésre kerestük a választ, hogy vajon milyen függvénynek felel meg az ábrázolt adatsor, azt találtuk, hogy az $1-(1-1/335)^k$ függvénnyel szinte teljesen megegyezik. Ez azt jelenti, hogy amikor a vizsgált függvény 1/1000-távol van a linearitástól, a POT egyetlen futtatása esetén az adatok alapján körülbelül 1/335 a valószínűsége annak, hogy nemlinearitást találunk (az elméleti eredményből következő 1/2000 valószínűség helyett).

7.2. Gráfok párosságának tesztelése

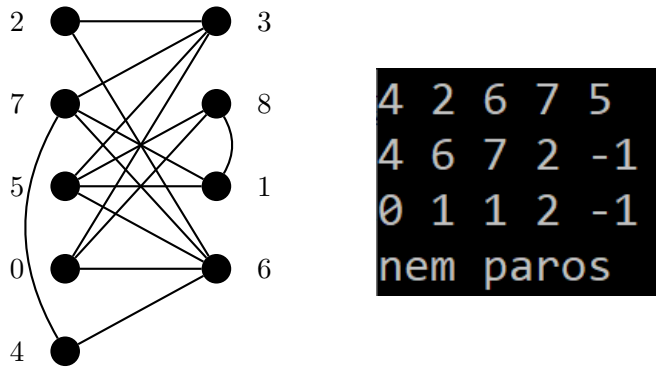
A programban egy véletlen gráfot generálunk a következőképpen. Először egy páros gráfot készítünk, amely színosztályainak méretei változtatható paraméterek. Az élek sűrűsége, azaz annak valószínűsége, hogy két, eltérő oldali csúcs között fut él, szintén megadható (ezek az élek Erdős–Rényi-modell-szerűen húzatnak be). Ezután a megadott távolságparaméter alapján $\epsilon N^2/2$ darab, a párosságot sértő élet húzunk be véletlenszerűen a két oldalra, a színosztályok méretarányának megfelelő számban. Az így kapott gráf tehát valójában $\epsilon/2$-távol van a párosságtól, az inkonzisztenciáért elnézést kérünk.

A sűrűmátrix-megadásnak megfelelően szomszédsági mátrixszal tároljuk a gráfot. Egy olyan publikus tagfüggvényen keresztül lehet kívülről hozzáférni a gráfhoz, aminek két egész típusú bemenő paramétere van, visszatérési értéke pedig 0 vagy 1: ez azt adja meg, hogy a két – sorszámmal reprezentált – csúcs között fut-e él a gráfban.

A helyesség ellenőrzésének, valamint az illusztrációk készítésének megkönnyítése érdekében egy olyan kódgeneráló függvény is a program részét képezi, ami a gráfot reprezentáló objektum alapján LaTeX kódot generál, ami a TikZ csomag használatával lehetővé teszi a gráf vizualizációját.

A fent leírt gráfból véletlenszerűen mintavételezünk néhány csúcsot, és a kapott feszített részgráfon a szélességi keresés (BFS – Breadth-First Search) algoritmusnak a párosság ellenőrzésére való változatát futtatjuk. Ezt többször megismételjük, és minden futásnál feljegyezzük, hogy találtunk-e a párosságot sértő élet.

A 7.3. ábrán egy, a program által generált kis méretű, 5+4 csúcsú gráf látható. Mellette egy öt elemű véletlen mintával történt futtatás eredménye szerepel. A futási eredmény első sorában a minta csúcsainak sorszámai szerepelnek; a második sorban ugyanezek, de már a gráf bejárása szerinti sorrendben; a harmadik sor pedig a megfelelő érték fölött a második sorban lévő csúcs BFS-fabeli mélységét tartalmazza. A második és harmadik sorban -1 érték jelzi, ha addig már nem jut el az algoritmus, mert felfedezte, hogy nem páros a gráf. Az utolsó sor a végeredmény: a vizsgált minta által feszített részgráf páros-e.

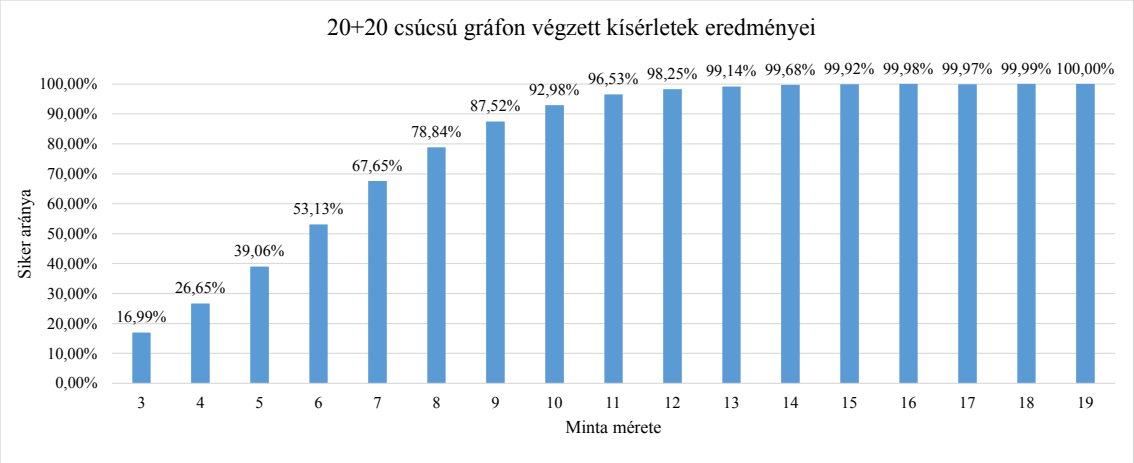


7.3. ábra. Egy generált gráf és egy hozzá tartozó futási eredmény.

Az ábrán látható példa esetén a különböző oldali csúcspárok között 0,6 valószínűséggel fut él, a távolságparaméter pedig $\epsilon = 0{,}05$. A futási eredményen követhető az algoritmus működése. Először az első sor első elemét, a 4-es csúcsot vizsgálja: ez a második sor első elme is, utána pedig a szomszédai következnek az első sorbeli sorrendjük szerint: 6 és 7. Az első csúcs mélysége 0, a szomszédaié 1. Ezután a következő, 6-os csúcs szomszédait keresi, amihez az első sort használja: a 4-et már vizsgálta; a 2 szomszéd, és az eddigiek alapján ez az él nem zavarja a párosságot ezért felírja a második sorba (2 mélységgel); a 6-ot épp most vizsgálja; a 7 is szerepel már, de még nem vizsgálta, és ezzel az éllel észreveszi a

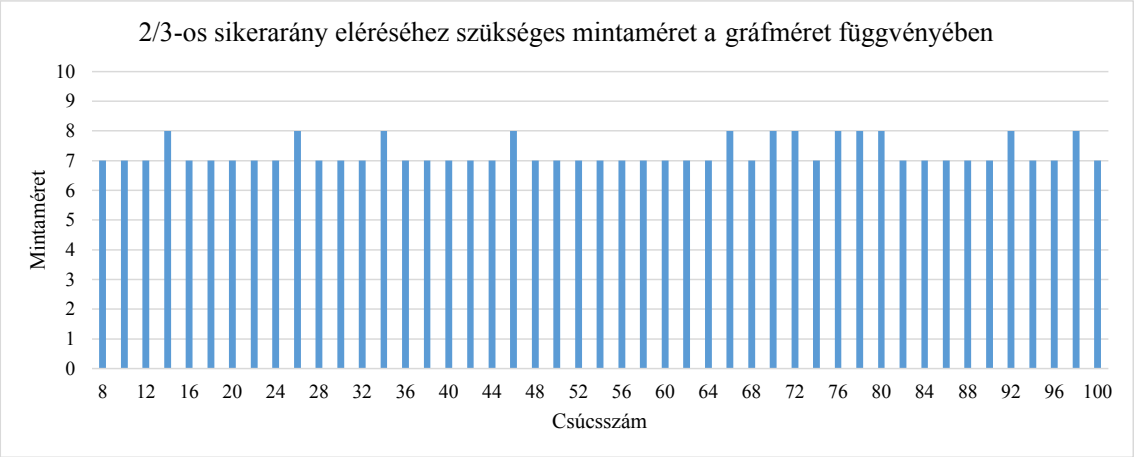
4-6-7 páratlan kört. Emiatt megáll a további vizsgálódás, már fel sem írja az 5-öt mint 6 szomszédját a második sor végére, csak kiírja, hogy nem páros a gráf.

A 20+20 csúcsú, a páros részen 0,6 sűrűségű gráfok esetén $\epsilon = 0{,}05$ paraméterrel 3-tól 19 méretű mintáig végeztünk kísérleteket, minden mintaméretre tízezerszer. Azt, hogy az egyes mintaméretek esetén a kísérletek hányadrészében jött rá az algoritmus, hogy a gráf nem páros, a 7.4. ábrán oszlopdiagramon ábrázoltuk.



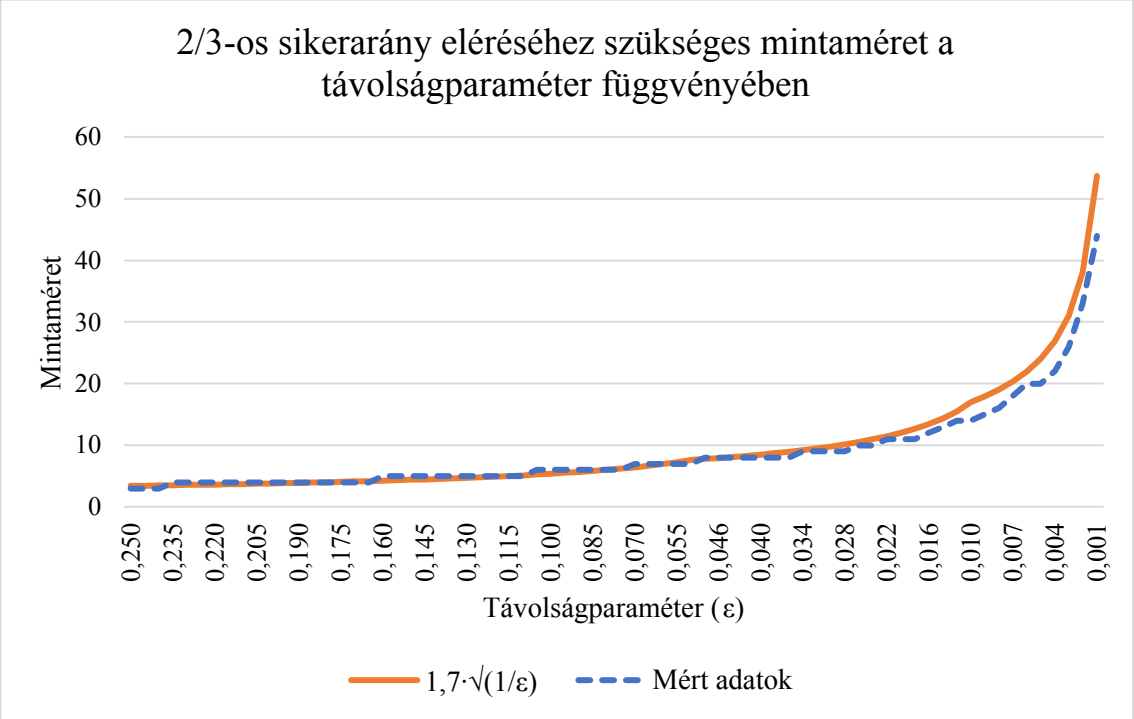
7.4. ábra. A tízezer futtatás eredményének összefoglalása.

Ugyanezt a kísérletet sokszor elvégeztük: 4+4-től kezdve 50+50 csúcsú gráfokig úgy, hogy a többi paraméter közben nem változott. Minden esetnél megnéztük, hogy legkevesebb hány csúcsból álló minta esetén éri el a nem-párosság felismerésének valószínűsége a 2/3-ot. Az eredmények a 7.5. ábra diagramján láthatók. Az elméleti eredményeknek megfelelően azt tapasztaljuk, hogy a gráf méretétől független ez az érték: mindig 7-8 körül mozog.



7.5. ábra. Különböző méretű (k+k típusú) gráfok esetén a 2/3-os elutasítási valószínűséghez szükséges mintaméret.

A távolságparamétertől való függést is kísérletekkel ellenőriztük, az erre vonatkozó eredmények a 7.6. ábrán láthatók. Az ϵ értéket 1/4-től kezdve 0,005-ös lépésekkel csökkentettük 0,05-ig; innen kezdve 0,002-es lépésekkel 0,01-ig; végül pedig 0,001-es lépésekkel 0,001-ig. Az eleinte használt 10+10 csúcsszám kicsi ϵ esetén már nem volt alkalmas a szükséges mintaméret követésére, ezért ekkor nagyobb gráfokkal végeztük a kísérleteket.



7.6. ábra. A távolságparaméter változásával a 2/3-os elutasítási valószínűséghez szükséges mintaméret. Figyelem, a vízszintes tengelyen ϵ értéke jobb kéz felé csökken.

A kísérleti eredmények mellett az $1{,}7\sqrt{1/\epsilon}$ függvényt is ábrázoltuk. Ez alapján úgy tűnik, hogy ahhoz, hogy 2/3 valószínűséggel felismerje az algoritmus a párosságtól ϵ-távoli gráfról, hogy nem páros, $O(\sqrt{1/\epsilon})$ méretű minta elegendő. Ezzel szemben a 4.1.3. szakaszban bemutatott és elemzett algoritmus $O(\log(1/\epsilon)/\epsilon^2)$ méretű mintát igényel. A 4.2.1. megjegyzésben már említettük, hogy ez összetettebb elemzéssel $\tilde{O}(1/\epsilon)$ csúcsra csökkenthető; ráadásul egy másik eredmény szerint nem-adaptív kérdések esetén mindenképp szükséges $\Omega(1/\epsilon)$ méretű minta. Ez arra utalhat, hogy bár a legtöbb gráf – de legalábbis a kísérletben használt típusú véletlen gráfok – esetén ugyan elegendő $O(\sqrt{1/\epsilon})$ méretű mintával dolgozni, viszont speciális, erre a célra konstruált gráfok esetén már $1/\epsilon$-nal arányos mintaméretet kapnánk. Ugyanis a véletlen gráfok struktúrája általában kifejezetten szabályos.

8. fejezet

Összegzés

A dolgozatban bemutattuk a tulajdonságtesztelés alapjait és kapcsolatát a PCP-elmélettel, valamint néhány függvény- és gráftulajdonság lokális tesztelésének legfontosabb elméleti eredményeit. Bizonyos tesztelő algoritmusokat viszonylag mélyrehatóan elemeztünk. Egy rövid kvantuminformatikai kitekintés után saját fejlesztésű programok segítségével bemutattuk, hogy az ismertetett elméleti eredmények nemcsak helytállóak, hanem a gyakorlatban általában jóval hatékonyabban is el lehet érni egy adott pontosságot.

A dolgozat legfontosabb eredményeit három pontban foglalhatjuk össze.

- A tárgyalt téma alapjainak, valamint néhány bonyolultabb eredménynek az értelmezése és bemutatása érthető formában, magyar nyelven.
- A feszített-C_4-mentesség tesztelésére szóló elméleti eredmény csekély mértékű javítása, pontosítása, valamint a benne szereplő konstansok konkrét értékeinek meghatározása.
- A linearitás teszt és a gráfok párosságát tesztelő algoritmus megvalósítása, ezek futtatásával tapasztalati adatok gyűjtése, azok feldolgozása, bemutatása és értelmezése.

A téma önmagában is szép és érdekes, ráadásul több alkalmazási területe is van.

Tulajdonságtesztelő algoritmusokkal akár hatalmas objektumok tulajdonságainak hatékony vizsgálata is lehetővé válik. Akkor is érdemes lehet teszteléshez folyamodni, ha az objektum gyorsan változik, és ezért lassabb algoritmus használatával, mire az eredményt megkapnánk, az már nem aktuális.

Napjaink kiterjedt hálózatainak vizsgálata hatalmas gráfok tulajdonságainak tesztelésével lehetséges – ha nem kívánjuk végigolvasni a nagyon hosszú (a gráfot leíró) bemenetet. Így sokkal gyorsabban kapunk eredményt, mintha pontos döntést hoznánk.

A függvénytulajdonságok hatékony vizsgálatát megvalósító algoritmusok sok fontos algebrai tulajdonság ellenőrzésének részét képezhetik. Ezenkívül például a PCP-tétel bizonyításában is fontos szerepet játszott mind a linearitás teszt, mind az alacsony fokú polinomok tesztelése.

Az önjavító kódok már a linearitás teszt eredeti [BLR93] cikkének címében is előkerülnek. Ezek lényege, hogy egy üzenetet olyan kódszavakkal kódolunk, amelyek valamilyen értelemben speciálisak, pl. pontokként reprezentálva egy egyenesre esnek. Így ha olyan kódot kapunk, ami nem esik egy egyenesre a többivel, akkor ezt kicserélhetjük az egyenesnek a hozzá legközelebb eső pontjára.

A PCP-elmélet első ránézésre teljesen elméletinek tűnik, de több gyakorlati alkalmazása is előfordul. Például a közelítő algoritmusok terén fontos eredmények születtek a téma fejlődésének melléktermékeként. Például ha $\mathbf{NP} \not\subseteq \mathbf{DTIME}(n^{\log\log n})$, akkor a MAXKLIKK problémára nincs konstans faktorú approximáció [FGL+91]. A terület a kódelmélet új ágát

hozta létre, a lokálisan tesztelhető kódokét (locally testable codes). Ennek lényege, hogy nem kell végigolvasni a kódszavakat, ha azok elég távol vannak egymástól, elég néhány helyen belenézni (mint egy PCP-bizonyítás esetén).

A kvantumalgoritmusok önmagukban is hatékonyabbá teszik bizonyos problémák megoldását, és ugyanez igaz a tulajdonságtesztelésre is. Ezért különösen érdekes a tulajdonságtesztelő kvantumalgoritmusok vizsgálata. A Bell-egyenlőtlenség alapján végzett kísérletek igazolták, hogy Einstein tévedett, a kvantumjelenségek nem írhatók le rejtett változók használatával.

További kutatás tárgya lehet egyrészt a tulajdonságtesztelő kvantumalgoritmusok területe, ahol új algoritmusokat lehet készíteni, vagy a korábbiakat hatékonyabbá tenni. Egy másik érdekes irány a klasszikus tulajdonságtesztelésen belül az ún. eloszlásmentes (distribution free) tesztelés, amely esetén nemcsak az egyenletes, hanem tetszőleges eloszlásból mintavételezhetjük például egy gráf csúcsait [Gol19].

Köszönetnyilvánítás

Elsősorban a konzulensemnek, Friedl Katalinnak szeretném megköszönni a türelmét és a rengeteg segítséget. Ráadásul a konzultációkat távolról kellett megoldani egyrészt a járvány, másrészt a svédországi tartózkodásom miatt. Köszönetet mondok továbbá Johan Håstadnak, a stockholmi KTH (Kungliga Tekniska Högskolan) tanárának, akinek 2020. őszi kurzusában több olyan téma is előkerült, ami bekerült a diplomamunkába. Végül a párizsi IRIF (Institut de Recherche en Informatique Fondamentale) igazgatójának, Frédéric Magniez-nek köszönöm, hogy felhívta a figyelmemet a téma néhány új kutatási irányára.

Irodalomjegyzék

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[ABRW16] Andris Ambainis, Aleksandrs Belovs, Oded Regev, and Ronald de Wolf. Efficient quantum algorithms for (gapped) group testing and junta testing. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, page 903–922, USA, 2016. Society for Industrial and Applied Mathematics.

[AF15] Noga Alon and Jacob Fox. Easily testable graph properties. *Combinatorics, Probability and Computing*, 24(4):646–657, 2015.

[AFKS00] Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. Efficient testing of large graphs. *Combinatorica*, 20:451–476, 05 2000.

[AFN07] Noga Alon, Eldar Fischer, and Ilan Newman. Testing of bipartite graph properties. *SIAM Journal on Computing*, 37:959–976, 01 2007.

[AK02] N. Alon and M. Krivelevich. Testing k-colorability. *SIAM J. Discret. Math.*, 15:211–227, 2002.

[ALM+92] S. Arora, C. Lund, R. Motwani, M. Sudan, and Mario Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, SFCS '92, pages 14–23, Washington, DC, USA, 1992. IEEE Computer Society.

[AS92] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs; a new characterization of NP. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, volume 45, pages 2–13. IEEE Computer Society, 11 1992.

[AS06] Noga Alon and Asaf Shapira. A characterization of easily testable induced subgraphs. *Combinatorics, Probability and Computing*, 15(6):791–805, 2006.

[AS07] Alp Atıcı and Rocco A. Servedio. Quantum algorithms for learning and testing juntas. *Quantum Information Processing*, 6(5):323–348, October 2007.

[Bab85] László Babai. Trading group theory for randomness. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, STOC '85, pages 421–429, New York, NY, USA, 1985. ACM.

[BFL90] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, SFCS '90, pages 16–25. IEEE Computer Society, 1990.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, STOC '91, pages 21–32, New York, NY, USA, 1991. ACM.

[BGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, STOC '88, pages 113–131, New York, NY, USA, 1988. ACM.

[BGS95] M. Bellare, O. Goldreich, and M. Sudan. Free bits, pcps and non-approximability – towards tight results. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, FOCS '95, page 422, USA, 1995. IEEE Computer Society.

[BHMT02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Quantum Computation and Information*, 305:53–74, 2002.

[Bla09] Eric Blais. Testing juntas nearly optimally. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 151–158, New York, NY, USA, 2009. Association for Computing Machinery.

[Bla10] Eric Blais. *Testing Juntas: A Brief Survey*, pages 32–40. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549 – 595, 1993.

[BT04] A. Bogdanov and L. Trevisan. Lower bounds for testing bipartiteness in dense graphs. In *Proceedings. 19th IEEE Annual Conference on Computational Complexity, 2004.*, pages 75–81, 2004.

[BV97] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM J. Comput.*, 26(5):1411–1473, 1997.

[CG04] Hana Chockler and Dan Gutfreund. A lower bound for testing juntas. *Information Processing Letters*, 90:301–305, 06 2004.

[Con91] Anne Condon. The complexity of the max word problem. In *Proceedings of the 8th Annual Symposium on Theoretical Aspects of Computer Science*, pages 456–465, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.

[Din05] Irit Dinur. A history of the PCP Theorem. 2005.

[ESY84] Shimon Even, Alan L. Selman, and Yacov Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159 – 173, 1984.

[FGL+91] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Approximating clique is almost NP-complete. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, SFCS '91, pages 2–12, Washington, DC, USA, 1991. IEEE Computer Society.

[FKR+04] Eldar Fischer, Guy Kindler, Dana Ron, Shmuel Safra, and Alex Samorodnitsky. Testing juntas. *Journal of Computer and System Sciences*, 68(4):753–787, 2004. Special Issue on FOCS 2002.

[GGR98] Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, July 1998.

[GMR85] S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304, New York, NY, USA, 1985. ACM.

[GMW86] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, SFCS '86, pages 174–187, Washington, DC, USA, 1986. IEEE Computer Society.

[Gol05] Oded Goldreich. On promise problems: In memory of shimon even (1935-2004). *Electronic Colloquium on Computational Complexity (ECCC)*, 2005.

[Gol10] Oded Goldreich. Introduction to testing graph properties. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:82, 01 2010.

[Gol17] Oded Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017.

[Gol19] Oded Goldreich. Testing graphs in vertex-distribution-free models. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 527–534, New York, NY, USA, 2019. Association for Computing Machinery.

[GR99] Oded Goldreich and Dana Ron. A sublinear bipartiteness tester for bounded degree graphs. *Combinatorica*, 19:335–373, 03 1999.

[GR04] Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32, 01 2004.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 212–219, New York, NY, USA, 1996. ACM.

[GS86] S Goldwasser and M Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, STOC '86, pages 59–68, New York, NY, USA, 1986. ACM.

[GS18] Lior Gishboliner and Asaf Shapira. Efficient Testing without Efficient Regularity. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54:1–54:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[GS19] Lior Gishboliner and Asaf Shapira. Efficient removal without efficient regularity. *Combinatorica*, 39:639–658, 2019.

[Hå01] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, July 2001.

[KKR04] Tali Kaufman, Michael Krivelevich, and Dana Ron. Tight bounds for testing bipartiteness in general graphs. *SIAM Journal on Computing*, 33, 02 2004.

[LFKN90] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, SFCS '90, pages 2–10 vol.1, Washington, DC, USA, 1990. IEEE Computer Society.

[MW16] Ashley Montanaro and Ronald de Wolf. *A Survey of Quantum Property Testing*. Number 7 in Graduate Surveys. Theory of Computing Library, 2016.

[PR02] Michal Parnas and Dana Ron. Testing the diameter of graphs. *Random Struct. Algorithms*, 20(2):165–183, March 2002.

[PRR06] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012 – 1042, 2006.

[RR14] Sofya Raskhodnikova and Ronitt Rubinfeld. *Linearity and Group Homomorphism Testing/Testing Hadamard Codes*, pages 1–6. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[RS96] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

[Sha90] Adi Shamir. IP = PSPACE. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, SFCS '90, Washington, DC, USA, 1990. IEEE Computer Society.

[Sze78] E Szemerédi. Regular partitions of graphs, colloque inter. *Proceedings, Colloque Inter.*, pages 399–401, 1978.

[Ver19] Rémi de Joannis de Verclos. Chordal graphs are easily testable. *arXiv: Combinatorics*, 2019.