BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM

Természettudományi Kar

Gráf lefedések és felhasználásuk szoftvertesztelésnél

Nyulasi Gergely

2017

Contents

1 Bevezetés

1. Bevezetés - Célkitűzés, kontextusba helyezés
2. Probléma bemutatása - bemutatod az automata tesztelés mi, miért jó, mi az ami nehéz
3. Formalizálás - a rendszer gráfos modelljének leírása
4. - Pont és útfedések - a modellen bemutatod hogy lehet a problémákat megmutatni pont és útfedésekkel
5. "Valódi példa" - ez a programozós rész, ahol rövid példán bemutatod
6. Összefoglalás - Tapasztalatok ,elvont következtetések, jövőbeni irány

2 Probléma bemutatása

Mindenki szeretné a feladatait minél hatékonyabban, optimálisan megoldani, hogy minél kevesebb idővel minél több feladatot tudjon megoldani. Megmutatjuk hogy a tesztelésnél is fontos ez a szempont és hogyan tudjuk elérni ezt, több különböző feladat esetén is.

2.1 Tesztelés alapjai, szoftvertesztelés

Manapság nagy hangsúlyt fektetünk a minőségre és annak biztosítására, mivel a minőséggel az értéket lehet reprezentálni és minél magasabb a minőség annál értékesebb az adott dolog, amit képvisel. A minőséget tesztelésekkel, próbákkal és egyéb módokon tudjuk ellenőrizni. Jelen esetben minket jobban érdekel a szoftvertesztelés és annak módjait járjuk körbe, de ezt majd később

látni is fogjuk.

A teszteléseknél megszokhattuk, hogy nem létezik maximális lefedettség, de minél nagyobb lefedettséget megpróbálhatunk elérni. Tesztelések során több különböző teszttel is találkozhatunk, melyek különböző lefedettségeket tekintenek fontosnak, ilyen például a smoke test, regressziós teszt, stb.

De mit is értünk tesztelés, tesztesetek alatt?
Mint korábban említettük a tesztelés a minőség ellenőrzésére szolgál és ezekkel könnyedén meg tudjuk mondani, hogy mennyire biztonságos, védett vagy éppen a célunknak megfelel.

Vegyük a teszteléssel kapcsolatos definíciókat az International Software Testing Qualification Board(ISTQB) alapján felállított megfogalmazások szerint. Ezek ismeretével jobb rálátás biztosítható a tesztelés mikéntjére és céljára.

Definíció 2.1 *Tesztelés: az összes szoftverfejlesztési életciklusokhoz kapcsolódó akár statikus, akár dinamukus folyamat, amely kapcsolatban áll a szoftvertermékek tervezésével, elkészitésével és kiértékelésével, hogy megállapitsa, hogy a szoftvertermék teljesiti-e a meghatározott követelményeket, megfelel-e a célnak. A tesztelés felelős a szoftvertermékkel kapcsolatos hibák megtalálásáért. [2]*

Definíció 2.2 *Teszteset: bemeneti értékek, végrehajtási előfeltételek, elvárt eredmények és végrehajtási utófeltételek halmaza, amelyeket egy konkrét célért*

vagy a tesztért fejlesztettek (például egy program forgatókönyv végrehajtása, vagy egy követelménynek való megfelelés). [IEEE 610 alapján]. [2]

Azaz a tesztelés alatt minél több hibát szeretnénk feltárni, megtalálni. Persze az, hogy ha találunk hibákat az nem jelenti, hogy minden hibát megtaláltunk. Nem könnyű megtalálni az összes hibát, vagy a mi esetünkben, amint majd látunk és részletesen be fogunk mutatni az összes élet egy gráfban. De hogy értsük is a kifejezéseket mostantól definiáljuk azokat is.

Most ismertetünk pár alapfogalmat, amit a kombinatorikában ismerünk és hasznosnak fogunk venni a bizonyítások és ismertetések során.

Ilyen például, hogy mit is értünk gráf alatt, vagy hogyan is képzeljük el vagy reprezentáljuk.

Definíció 2.3 *A $G = (V, E)$ pár egy egyszerü gráf, ha $V \neq \emptyset$ és $E \subseteq V2 := u, v : u, v \in V, u \neq v$, azaz E elemei V bizonyos kételemü részhalmazai. Ha G egy gráf, akkor $V(G)$ az a V halmaz, és $E(G)$ az az E halmaz, amire $G = (V, E)$. A G egyszerü gráf véges, ha V véges halmaz. [4]*

Definíció 2.4 *A G gráf egy diagramja a G egy olyan lerajzolása, amiben a csúcsoknak (síkbeli) pontok felelnek meg, éleknek pedig olyan síkgörbék, amelyek az adott él két végpontját kötik össze, önmagukat nem metszik, és más végpontokat elkerülnek. Az $e = u, v$ élt röviden $e = uv-$vel jelöljük, $u-t$ és $v - t$ az e él végpontjainak mondjuk. Az u és v csúcsok szomszédosak, ha $uv \in E$. Az e és f éleket párhuzamosnak nevezzük, ha végpontjaik azonosak. Hurokél az olyan él, aminek két végpontja megegyezik. A $G = (V, E)$ pár*

gráf, ha $V \neq \emptyset$, E élhalmaz $V - n$, és párhuzamos és hurokél is megengedett. [4]

Igy a kombinatorikai tanulmányaink segítségünkre lesznek, hogy minél pontosabban tudjuk modellezni a tesztelések menetét, illetve hogy azt könnyebben elmagyarázhassuk.
Mint már említettük a különböző teszteseteket tekinthetjük a kombinatorikában vett gráfokkal és azok élsorozataival.

Vegyük most azt, hogy mit is értünk élsorozaton és hozzá kapcsolódó definíciókat:

Definíció 2.5 *A G gráf élsorozata egy olyan $(v_1, e_1, v_2, e_2, \cdots, v_k)$ sorozat, amire $e_i \in E(G)$ és $e_i = v_i v_{i+1} (\forall 1 \leq i < k)$. A séta olyan élsorozat, aminek minden éle különböző. A körséta olyan séta, aminek kiinduló és végpontja azonos: $v_1 = v_k$. AZ út (ill. kör) olyan (kör)séta, aminek csúcsai (a végpontok azonosságától eltekintve) különbözők.*
Egyszerü gráf esetén az út (kör) azonositható a hozzátartozó pontsorozattal vagy élsorozattal. [4]

Definíció 2.6 *A G gráf összefüggő(öf), ha bármely két pontja között vezet séta. [4]*

Definíció 2.7 *A G gráfban pontosan akkor létezik u és v között séta, ha létezik u és v között út. [4]*

További bejárások is lehetségesek, amelyek

1. élekre szól(Euler), valamint

2. csúcsokra szól(Hamilton).

Ezek után a következő definíciókat lehet megállapítani:

Definíció 2.8 *A $G = (V, E)$ gráf Euler-sétája (Euler-körsétája) a G gráf egy olyan (kör)sétája, amely G minden élét (pontosan egyszer tartalmazza). [4]*

Tétel 2.1 *Ha a $G = (V, E)$ gráf véges és összefüggő, akkor*

1. *G-nek pontosan akkor van Euler-körsétája, ha G minden csúcsa páros fokú, ill.*

2. *G-nek pontosan akkor van Euler-sétája, ha G-nek 0 vagy 2 páratlan fokú csúcsa van.*

[4]

Ha élek helyett csúcsokról beszélünk, akkor egy másik fontos fogalomhoz jutunk el.

Definíció 2.9 *A G gráf Hamilton-köre (Hamilton-útja) a G olyan köre (útja), mely G minden csúcsát tartalmazza. [4]*

Mivel egy körben (útban) szereplő minden csúcs különböző, ezért a Hamilton-kör (Hamilton-út) a G gráf olyan bejárása, mely G minden csúcsát pontosan egyszer érinti.

Definíció 2.10 *Ha a véges G gráfban létezik Hamilton-kör (ill. Hamilton-út), akkor G-nek k tetszőleges pontját törölve, a keletkező gráfnak legfeljebb k (ill. k+1) komponense van. [4]*

Ezek a definíciók sajnos számunkra csak kis mértékben hasznosak. Mivel nekünk fontos az is hogy több különböző séta is áthaladhasson ugyanazon a ponton is vagy többször használhassunk egy élet, ilyen például, ha ugyanabba a pontba érkezünk, de több elágazás van amin szeretnénk elindulni, vagy ilyen ha egy élből több él is indul ki, amelyek egyenlő fontosak és csak az előbbi élből tudunk indulni.

Majd folytassuk a teszt, állapotátmenetek és scenario-k megértésével, mivel a fentiekben már kifejtettünk pár definíciót, ami segítségünkre van:

Definíció 2.11 *Teszt: egy vagy több teszteset halmaza. [IEEE 829]. [2]*

Minél több teszt sikeres egy adott tesztelés során, annál Most hogy már a teszt fogalma világos, ekkor a szoftver fogalmát definiáljuk:

Definíció 2.12 *Szoftver: számitógépes programok, folyamatok és esetlegesen a számitógépes rendszer üzemelésére vonatkozó dokumentációk és adatok. [IEEE 610]. [2]*

Továbbiakban az automatizálásról lesz szó, és annak hasznosságáról.

2.2 Automata szoftvertesztelés

Mit is értünk automatizálás alatt? Olyan folyamat, amikor technológiák alkalmazása során az emberi munkát csökkentjük, hogy minél alacsonyabb legyen a humán erőforrás igénybevétele. Persze több erőforrást vesz igénybe és több tudást az elején, de a többszöri futások miatt, azonban idővel megtérül a bele fektetett idő és munka. Igy hosszabb távon megéri automatizálni a teszteket,

főleg ha hosszabbak, vagy bonyolultabbak.
Automatizálás mellett szól: a termelékenység növelése, a költségek csökkentése hosszú távon, és nem utolsósorban a minőség javítása és az alacsonyabb hibaszázalékok elérése.
Az automatizált szoftver tesztelés definíciója a következő:

Definíció 2.13 *Teszt automatizálás: valamilyen szoftver használata különböző teszttevékenységek támogatására, mint például tesztmenedzsment, müszaki teszttervezés, tesztek végrehajtása, teszteredmények vizsgálata. [2]*

Azaz bizonyos szoftverek segítségével teszteseteket hajtunk végre hogy nekünk ne kelljen manuálisan(humán erőforrás csökkentése) elvégezni őket. Ebben az esetben gráfok bejárását fogjuk érteni

Definíció 2.14 *Feltételezzük, hogy a tesztelendő rendszer különböző rendszer állapotokkal rendelkezik, melyek között nem szabad az átjárás. Ilyen állapot lehet például a bejelentkező képernyő, ahonnan át lehet menni a user control panelre vagy az elfelejtett jelszó képernyőre. User control panel $->$ elfelejtett jelszó átmeneti viszont nem létezik. Azaz ezeket a rendszer állapotokat tekinthetjük egy gráf különböző csúcsainak, valamint a köztük lévő élek lesznek azon relációk, hogy melyik állapotból érhetjük el a másikat. Ezt a gráfot vegyük mostantól G gráfnak.*

Definíció 2.15 *Állapotátmenet: átmenet egy komponens vagy rendszer két állapota között. [2]*

Definíció 2.16 *A rendszer állapotok közötti átmenetek közül annak a kiválasztását, hogy melyik történik meg vagy a kapott input, vagy valamilyen sztochasztikus*

folyamat szabályozza. Tehát a G gráf kezdő (S) csúcsát és abból való elágazásokat vesszük.

Definíció 2.17 *Vannak teszt esetek(scenario-k) amik ilyen állapotok és átmenetek(inputok) sorozatát tartalmazzák. Például: Áru lekérés scenario: bejelentkezés − > kezdőképernyő − > áru lekérés. Legyenek ezek a G gráfon vett különböző utak. Azért nem diszjunktak ezek az utak mert két útnak lehetnek közös éleik, viszont el fognak térni egymástól.*

Definíció 2.18 *Vannak teszt forgatókönyvek, amik több scenario egymás utáni lefutását tartalmazzák. Vegyük ezt a G gráfon értett utak halmazának, legyen ez mostantól H.*

Megjegyzés 2.1

1. *Egy teszt forgatókönyv akkor helyes, ha minden benne lévő teszt eset "le tud futni", azaz az előző eset olyan állapotban ér véget ahonnan az új el tud indulni. Azaz minden utat amit H tartalmaz, azokon végig tudunk sétálni.(meg lehet csinálni?)*

2. *Ez alapján a teszt forgatókönyv több út a gráfban.*

Ezeket a rendszer állapotokat, állapotátmeneteket nézhetjük úgy is mint egy gráf csúcsai vagy élei, attól függően hogy számunkra mi a kényelmes, vagy hogyan egyszerűbb elképzelni, modellezni. Viszont arra oda kell figyelni, hogy ha az állapotokat csúcsoknak nézzük, akkor egyes állapotokból más állapotokba lehet eljutni, de nem feltétlen oda-vissza lehetséges ez, ezért nézzük a gráfokat irányított gráfoknak.
Ezeket a következőképp tudjuk elképzelni:

Definíció 2.19 *$D = (V, A)$ irányitott gráf, ha $V \neq \emptyset$ és $A \subseteq V^2$. (Minden élnek van egy irányitása. A diagramon nyilakkal szokás jelölni. Párhuzamos és hurokél itt is értelmezhető, és akár mindkét irányú él be lehet húzva két pont között. Az irányitatlan fogalmak jó része értelemszerüen kiterjed.) [4]*

Definíció 2.20 *Egy G (irányitott) gráf aciklikusnak mondunk, ha G nem tartalmaz (irányitott) kört. [4]*

Ezek után szeretnénk olyan gráf bejárást találni, amivel az összes csúcsot vagy összes élet le tudjuk fedni. Ezt a két esetet többféle módon meg tudjuk oldani algoritmusokkal. Viszont a lehető legkevesebb úttal megoldani ezt már nem olyan egyszerű.
Ennek az optimalizálására keresünk megoldást a következőkben. Valamint ehhez az eléréshez tudnunk kell további definíciókat, és hogy miért is keresünk ilyet? Ilyen pont- és élfedéseket regressziós-, rendszer- vagy smoke tesztelésnek is nevezhetünk. De ezek mind különböző dolgokat jelentenek és fednek le.

Definíció 2.21 *Regressziós teszt: egy korábban már tesztelt program módositást követő tesztelése annak biztositása érdekében, hogy a módositás nem okozott hibát a szoftver nem módositott részeiben. A tesz végrehajtása a szoftver vagy a szoftverkörnyezet változásakor történik. [2]*

Definíció 2.22 *Rendszerteszt: integrált rendszer tesztje abból a célból, hogy ellenőrizzük a követelményeknek való megfelelőségét.[Hetzel]. [2]*

Definíció 2.23 *Smoke teszt: A definiált, illetve tervezett tesztek egy olyan halmaza, amely komponens, illetve a rendszer fő funkciónalitásait hivatott tesztelni abból a célból, hogy meggyőződjünk arról, hogy a program legkritikusabb részei müködnek-e. A teszt során nem megyünk bele a részletekbe. A napi integráció és a smoke tesztek a leggyakrabban használt ipari eljárások közé tartoznak. [2]*

Definíció 2.24 *Állapot-tábla: Egy táblázat, ami minden egyes állapotra és lehetséges eseményre mutatja az állapotátmeneteket, megjelenitve az érvényes és érvénytelen átmeneteket is. [2]*

Ezek a főbb tesztek, amelyeket tekinteni fogunk és megnézzük, hogy tudjuk-e esetleg modellezni őket.

3 Alap ötlet

3.1 Gráf alapismeretek

A gráfokon való lefedést többféleképpen lehet nézni. Egyrészt pontok lefedése-, valamint élek lefedéséről lehet beszélni. Ezek alkalmazhatóak különböző feladatok megoldására vagy optimalizálásra.
Az útlefedést vissza tudjuk vezetni a páros gráfokban alkalmazott maximális párosítás keresésre.

Nézzük meg mit is jelent az hogy egy gráf páros, de előtte meg kell még értenünk azt is hogy mit értünk gráf színezésen is.

Definíció 3.1 *A G gráf k szinnel szinezhető, ha G minden csúcsa kiszinezhető k adott szin valamelyikére úgy, hogy G bármely élének két végpontja különböző szinü legyen. A G gráf kromatikus száma $\chi(G) = k$, ha G kiszinezhető k szinnel, de k-1 szinnel még nem. [4]*

Amikor egy gráf *kiszinezéséről* beszélünk (hacsak nem jelezzük az ellenkezőjét), mindig a csúcsoknak a fenti szabály szerinti színezésére gondolunk. Egy konkrét színezés esetén az azonos színűre festett csúcsok halmazát (amely halmaz tehát nem feszíthet élt) *szinosztálynak* nevezzük. Jegyezzük meg, hogy a színosztály mindig a színezéstől függ, és általában nem egyértelmű, hogy egy G gráfot hogyan is kell $\chi(G)$ színnel kiszínezni.

Definíció 3.2 *A G gráf páros gráf, ha G két szinnel kiszinezhető, azaz, ha $\chi(G) \leq 2$. [4]*

Megjegyzés 3.1 *A fenti definíció azzal ekvivalens, hogy a G gráf pontosan akkor páros, ha G csúcsai két diszjunkt halmazba oszthatók úgy, hogy G minden éle a két halmaz között fut, azaz mindkét halmazban van egy-egy csúcsa.(Ez egyébként a páros gráf definíciója.) Minden páros gráfnak van tehát két színosztálya, amelyek között az élei futnak. Azonban ez a két színosztály nem feltétlenül egyértelmű: például az n pontból álló üres gráf csúcsainak tetszőleges két osztályra bontása teljesíti a feltételt.*

Ha már tudjuk hogy mit jelent a páros gráf akkor meg tudjuk határozni a párosítást valamit a maximális párosításokat egy gráfban.

3.2 Maximális párositás

Definíció 3.3 *Adott G gráf esetén $\nu(G)$ jelöli a G független élhalmazai közül a maximális méretét, azaz G maximális párositásának elemszámát. [4]*

Valamint a részben rendezett párokat vehetjük speciális hálózatok típusainak, és megannyi optimalizálási problémát meg lehet oldani folyam módszerrel, amit részben rendezett párokon találtak. Alap tétel a Dilworth-féle minimum-maximum reláció a maximális nagyságú antiláncra.

Legyen $(S, \leq)$ pár egy részben rendezett pár, ahol S a pár és a $\leq$ a reláció, ami kielégíti S követelményeit:

1. $s \leq s$ (reflexív),
2. ha $s \leq t$ és $t \leq s$ akkor $s = t$ (antiszimmetrikus),
3. ha $s \leq t$ és $t \leq u$ akkor $s \leq u$ (tranzitív),

ahol minden $s, t, u \in S$. Használjuk az $s < t$ kifejezést, ha $s \leq t$ és $s \neq t$ is teljesül. Jelenleg csak a *véges* részben rendezett párokat, azaz S véges. Legyen $C \in S$ olyan részpár, amit *láncnak* nevezünk, ha $s \leq t$ vagy $t \leq s$, ahol minden $s, t \in C$. Legyen $A \in S$ olyan részpár, amit *antiláncnak* nevezünk, ha st és ts teljesül minden $s, t \in A$-ra. Ennél fogva ha C lánc és A antilánc akkor

$$|C \cap A| \leq 1.$$

Ebből következnek az alábbi állítások. [1]

Definíció 3.4 *$s_1 \leq s_2 \leq \cdots \leq s_k$ lánc, ha bármely két eleme relációban áll. $s_1, s_2, \cdots, s_k$ antilánc, ha semelyik két eleme nem áll relációban. [1]*

Tétel 3.1 *Legyen $(S, \leq)$ egy részben rendezett pár. A minimális antiláncok száma ami lefedi S-t megegyezik a maximális láncok számával. [1]*

Tétel 3.2 *Legyen $D = (V, A)$ egy aciklikus irányított gráf. A minimális irányított vágások száma, ami lefedi A-t megegyezik a maximális irányított út hosszával. [1]*

Tétel 3.3 *(Dilworth-féle felbontási tétel) Legyen $(S, \leq)$ egy részben rendezett pár. Ekkor a minimális láncok száma ami lefedi S-t megegyezik a maximum antiláncok számával. [1]*

Tétel 3.4 *Legyen $D = (V, A)$ egy aciklikus irányított gráf. Ekkor a minimális út lefedések száma ami lefedi az összes élet megegyezik a maximális száma*

azoknak az éleknek amelyek semelyik kettő nincs összekötve az irányított útban. [1]

Valamint a húrok fedésére tudjuk, hogy:

Tétel 3.5 *Legyen $D = (V, A)$ egy aciklikus irányított gráf. Ekkor a minimális útfedési húrok száma megegyezik a maximális irányított vágások számával. [1]*

Tétel 3.6 *Legyen $D = (V, A)$ egy aciklikus irányított gráf, aminek pontosan egy kiindulási pontja van, s, és pontosan egy érkezési pontja van, t. Ekkor a minimális $s - t$ útfedések száma A-ban megegyezik a maximális irányított $s - t$ vágások számával. [1]*

Tétel 3.7 *Legyen $D = (V, A)$ egy aciklikus irányított gráf és $B \subseteq A$. Ekkor a minimális útfedések száma B-ben megegyezik a $|C \cap B|$ maximumával, ahol C az irányított vágások számával. [1]*

4 Pont- és Útfedési algoritmusok

4.1 Útlefedési algoritmus gráfon

Vegyük G gráfot, aminek a csúcsai a következők: van egy s csúcs, $v_1, v_2, \cdots, v_n$ csúcshalmazok és egy t csúcs. Az élek az s-ből v_1-be, v_n-ből t-be, valamint minden i-re v_i és v_{i+1} csúcsai között futnak, de nem feltétlenül minden csúcs között.
Irányítsuk így az éleket, ahogy írjuk, s-ből ki t-be be és v_i-ből v_{i+1} felé. Minden pontba vezet él, s kivételével. Minden pontból is vezet él, t kivételével. Legyen ekkor ez a G gráf irányított gráf.
Ekkor elegendő azt vizsgálnunk, hogy lefedjük az összes csúcsot a lehető legkevesebb irányított $s-t$ úttal.
Majd vegyük G'-t mint G tranzitív lezártját. Ekkor egy részben rendezett halmazt kapunk.

A tranzitív lezárt fogalmát a következőképp definiáljuk:

Definíció 4.1 *Egy $G = (V, E)$ gráf tranzitív lezárása $G' = (V', E')$ gráf, ahol $V' = V$ és $(u, v) \in E'$, akkor és csak akkor, ha létezik $u- > v$ út a gráfban.*

A fentiek alapján, ha egy $G = (V, E)$ gráf tranzitív lezártja G' gráf, akkor azt úgy tudjuk ábrázolni, hogy $u - ból$ húzunk egy élt $v - be$ akkor és csak akkor ha $u - bólv - be$ el tudunk jutni.

Ekkor beláthatjuk hogy ebben a gráfban pontosan ugyanannyi útra van

szükség a gráffedéshez, mint az eredetiben.

Majd ebben a G illetve G' gráfban minden v csúcsból csinálunk egy $v_{alsó}$ és $v_{felső}$ csúcsot. Ekkor összekötjük $v_{alsó}$-t $u_{felső}$-vel akkor és csak akkor, ha u nagyobb mint v. Beláthatjuk, hogy ha az egész részben rendezett halmaz, vagyis G' egy lánc, akkor van olyan párosítás ebben a páros gráfban, ami két pont kivételével mindent bepárosít. Hasonlóan, a részben rendezett halmaz felbomlik k lánc uniójára akkor és csak akkor ha van olyan párosítás, ami $2k$ pont kivételével mindent bepárosít.

Tehát keresni kell egy maximális párosítást, és az megmondja, hogy mi a minimális láncfelbontás. Sőt, magukat a láncokat is megtalálhatjuk ebből.

4.2 Egyéb algoritmusok

Vegyünk további algoritmusokat, melyekkel maximális párosítást tudunk keresni gráfokban, hogy jobb rálátásunk legyen a több lehetséges megoldásra is.

4.2.1 Magyar módszer

Az eljárás alapgondolata az, hogy egy tetszőleges párosításból kiindulva ismételten javítható utakat keres egészen addig, amíg ilyen létezik. (Egy olyan utat, ami párosítatlan A-beli pontból indul és minden második éle az aktuális párosításhoz tartozik, alternáló útnak hívjuk.) Ha talál egy javító utat, akkor az ebben lévő, aktuális párosításhoz tartozó éleket kidobja a párosításokból, az úton lévő, párosításban nem szereplő éleket pedig beveszi a párosításba; így a párosítás élszáma eggyel nő. Ha pedig már nincs több javító út, akkor egy egyszerü tétel garantálja, hogy az aktuális párosítás maximális. A bizonyítás alapgondolata a következő: ha az algoritmus már leállt és az M párosítást adta, akkor jelöljük U-val az M által le nem fedett A-beli pontok halmazát, T-vel az U-ból alternáló úton elérhető B-beli pontok halmazát, T-vel pedig a T'-beli pontok M szerint vett párjait. Ekkor könnyen belátható, hogy minden $(T \cup U)$-beli pont minden szomszédja T'-ben van. Ebből pedig következik, hogy $T' \cup (A \leq (T \cup U))$ ugyanolyan méretü lefogó ponthalmaz G-ben, mint M éleinek száma, így M-nél nagyobb párosítás nem lehet. [3]

4.2.2 Alternáló utas algoritmus

A Kőnig tétel iménti bizonyitásából hatékony algoritmust kaphatunk egy páros gráf maximális párositásának ill. minimális lefogó ponthalmazának megtalálására. Ha ugyanis a maximális folyamok meghatározására szolgáló javitó utas módszert a Kőnig tétel bizonyitásában leirt konstrukcióra alkalmazzuk, és eltekintünk az s-re ill. t-re illeszkedő élektől, akkor az alábbi eljárás adódik. Kiindulunk az üres párositásból, és azt javitgatjuk. Ha már találtunk egy M párositást, akkor tekintjük az M-hez tartozó segédgráfot, azaz M éleit B-ből A-ba irányitjuk, G egyéb éleit pedig A-ból B-be.

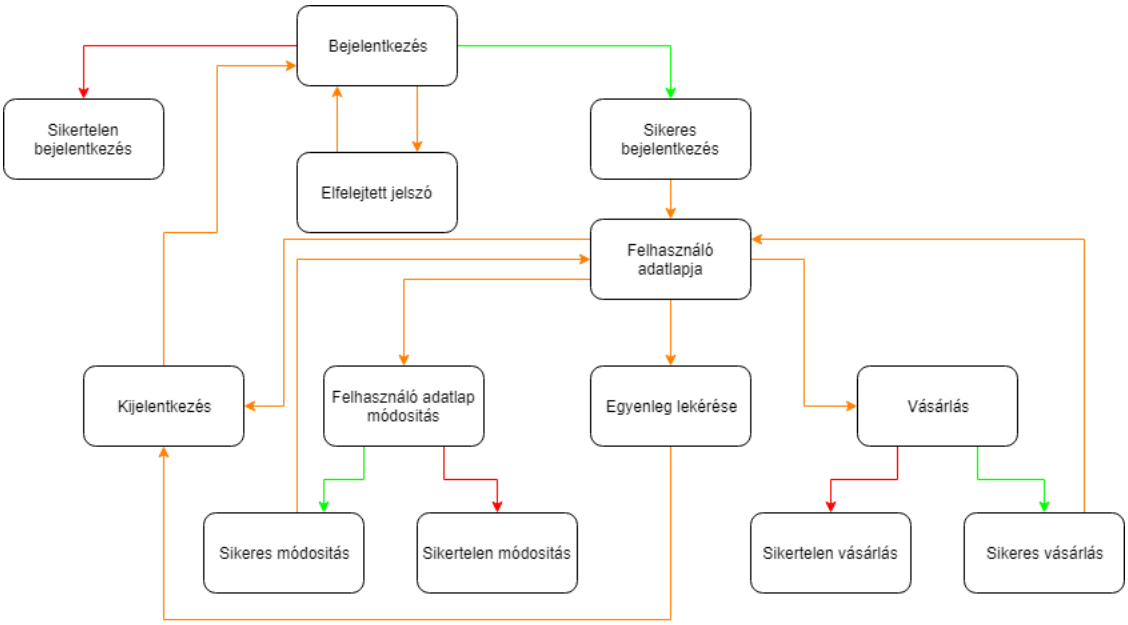
Ha ebben a segédgráfban létezik egy P irányitott út egy A-beli, az aktuális M párositás által fedetlen pontból olyan B-beli pontba, melyet szintén nem fed a párositás, akkor ezen az úgynevezett alternáló úton az eddigi párositáséleket elhagyva, és P párositáson kivüli éleit bevéve (más szóval M helyett $M \Delta P$-t tekintve), egy eggyel nagyobb méretü párositást kapunk.

kép

Ha pedig nincs javitó alternáló út, akkor M maximális párositás, és könnyen található egy —M— csúcsot tartalmazó lefogó ponthalmaz is.

5 Példa

5.1 Példa gráf



5.2 Algoritmus alkalmazása

6 Konklúzió

References

[1] Immanuel M Bomze, Marco Budinich, Panos M Pardalos, and Marcello Pelillo. The maximum clique problem. In *Handbook of combinatorial optimization*, pages 217–224. Springer, 1999.

[2] Magyar Szoftvertesztelői Tanács Egyesület. Szoftvertesztelés egységesitett kifejezéseinek gyüjteménye, 2014.

[3] A Frank. A magyar módszer és általánosıtásai,[in hungarian: The hungarian method and its extensions] szigma. *XXXIII (1–2)*, pages 13–44, 2002.

[4] Fleiner Tamás. A számitástudomány alapjai, 2014.