Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

Trautsch László Kálmán

AUTOMATIZÁLT ZÁRÓVIZSGABEOSZTÁS-KÉSZÍTÉS ÁLLAPOTTERÉNEK CSÖKKENTÉSE GRÁF-ALAPÚ MODELLEL

KONZULENS

Erdős Szilvia

BUDAPEST, 2021

Tartalomjegyzék

Összefoglaló

Az automatizált beosztástervezés évtizedek óta kutatott téma, amely napjainkban is számos kihívást rejt. Ennek részfeladata a záróvizsgabeosztás-készítés, amelynek során speciális követelményeket és hatalmas méretű lehetséges állapottereket kell kezelni.

Dolgozatomban beosztástervezési feladatok állapotterét csökkentő, saját gráf-alapú módszeremet mutatom be a Budapesti Műszaki és Gazdaságtudományi Egyetem (BME) 100 fős heterogén hallgatói csoportja záróvizsgabeosztás-készítésének problémáján keresztül. A különféle szigorúsági szintű, változatos követelmények megadására saját modellt használok, amelynek segítségével az algoritmus lecsökkenti a probléma állapotterét, vagyis ellentmondásos, a szigorú szabályoknak nem megfelelő állapotokat zár ki. Továbbá a szabályos állapotokhoz minimális és maximális büntetőpontszámokat számít. Az állapottér-csökkentés különböző heurisztikák, megoldóalgoritmusok alapjául szolgálhat, vagy folyamatosan is alkalmazható azok használata során. Így teljesen kizárhatóak a szigorú követelményeknek nem megfelelő, esetleg nagyon magas büntetőpontszámmal ellátott beosztások, továbbá az is kiderül, ha már a kezdeti adatokban ellentmondások vannak.

Az eredményeim mutatják, hogy a módszer egy ilyen komplexitású probléma kezelése során az összes szigorú és gyenge szabállyal tud dolgozni, valamint az állapotteret jelentős mértékben képes csökkenteni. A modell továbbfejlesztve a jövőben tetszőleges beosztástervezési feladatokat megoldó algoritmusoknál alkalmazható lehet alapként.

Abstract

Automated scheduling is a topic that has been researched for decades and is still challenging today. A subtask of this is final exam scheduling, in which special requirements and huge possible state spaces shall be managed.

In this dissertation, my own graph-based method for reducing the state space of scheduling tasks is presented through the problem of final exam scheduling for a heterogeneous group of 100 students of Budapest University of Technology and Economics (BME). I use my model to specify various constraints with different levels of strictness, which can be used by the algorithm to reduce the state space of the problem, thus excluding states that are contradictory and do not comply with the hard constraints. Furthermore, it calculates the minimum and maximum penalty points for the valid states. The state space reduction can be used as a basis of various heuristics, solving algorithms, or can be applied continuously while they are being used. In this way, schedules that do not comply with hard constraints or receive too many penalty points can be excluded entirely, and any inconsistencies in the input data can be detected.

The results show that this method can work with all the hard and soft constraints when handling a problem of such complexity and can reduce the state space significantly. This model can be improved and used as a base for future algorithms solving any kind of scheduling problems.

1 Bevezetés

A záróvizsgabeosztás-készítés jelenleg egy kézzel készített, hosszadalmas folyamat, amely során számos követelményt kell figyelembe venni, és azok alapján megtalálni a lehető legoptimálisabb beosztást. Mivel manuálisan készítik, ezért számos hibalehetőség van, és nehéz átlátni, mennyire sikerül jól egy elkészített beosztás.

Épp ezért az automatizálásra számos kísérlet született, azonban a rendkívül sok lehetséges beosztás, tehát a hatalmas méretű állapottér, továbbá a változatos és gyakran egymásnak is ellentmondó szabályok miatt ez még napjainkban is rengeteg kihívást rejt magában.

Dolgozatomban a változatos követelmények kezelésére, valamint az állapottér méretének csökkentésére keresek megoldást.

1.1 A problémakör

A záróvizsgabeosztás-készítés a beosztástervezési problémák egy részfeladata, ahol speciális követelményeket és hatalmas méretű lehetséges állapottereket kell kezelni.

Dolgozatomban a Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar szóbeli záróvizsgájával foglalkozom, ennek leírását az egyetemi Tanulmányi és Vizsgaszabályzat [1], valamint annak kari kiegészítése [2] tartalmazza. Az alfejezetben a problémakör áttekintését mutatom be, a konkrét szabályokat a 4. fejezetben ismertetem részletesen.

1.1.1 A záróvizsgabeosztás-készítés áttekintése

A záróvizsgabeosztás-készítés célja a hallgatók besorolása vizsgaalkalmakra. A vizsgaalkalmaknál szükség van elnökre, titkárra, belső tagra, konzulensre és a hallgató vizsgatárgyaiból oktatókra. A szerepek között vannak összevonhatóak, amelyek közül többet is betölthet egyszerre egy oktató. De nem akármelyik szerepek vonhatóak össze.

A vizsgákat időpontokra is be kell osztani, ezt 5 perces pontossággal vizsgáljuk. Lehetséges párhuzamosan több teremben is vizsgáztatás, de ezeknek a számát minimalizálni kell.

A vizsgák blokkokat alkotnak, a blokkok pedig egész napos szekciókat. Ezek közül mindegyiknek eltérő lehet a hossza. Követelmények szabályozzák, hogy a különböző hosszok, kezdési és végidőpontok mennyire optimálisak.

Feltételek vannak az oktatók blokkokon és egész napos szekción történő részvételére is. Továbbá ebédszüneteket is kell tartani, és a beosztásban ezeket is szerepeltetni kell. Az ebédszünetek is különböző időpontokban és különböző időtartamúak lehetnek.

A hallgatók képzését és képzési szintjét is figyelembe kell venni. Például az oktatókhoz meg van adva, hogy mely képzésekből tarthatnak vizsgát, továbbá egy blokkon belül minimalizálni kell a vizsgák képzésszintváltásának számát.

Az oktatókról elérhető órás bontásban, hogy mely időpontokban érnek rá vizsgáztatni, és mikorra nem oszthatók be. További cél az oktatók terhelésének kiegyenlítése az egyes szerepkörökön belül, és az összes terhelés minimalizálása.

Szabályok vannak még az egyes vizsgáztatási szekciók és oktatók folytonos beosztására nézve is.

A különböző megengedhető, de nem optimális társításokhoz büntetőpontszámok rendelhetők, aminek segítségével szabályozható, melyik követelményt mennyire fontos betartani. A cél olyan beosztás készítése, ahol minimális az összes büntetőpontszám.

1.1.2 A probléma komplexitása

A rengeteg és változatos szabály, valamint a beosztások lehetséges száma alapján látható a feladat komplexitása és az állapottér hatalmas terjedelme.

További nehézséget jelent az összes kritérium pontos formalizálása is. Számos nehezen leírható követelmény van, többek között az oktatók és termek ebédszüneteinek kezelése, a különböző szekciók folytonosságának és párhuzamos zajlásának biztosítása, és az egy blokkon belüli képzésszintváltások számának minimalizálása. Vannak egymásnak ellentmondó feltételek is, például az oktatók terhelésének kiegyenlítése amikor több szerepet is betölthetnek, vagy kevés olyan időpontot adtak meg, amikor rá tudnak érni. Az is előfordulhat, hogy egy hallgató konzulense biztosan más időpontokban ér rá, mint amikor a hallgató a többi követelmény alapján vizsgáztatható lenne. Maguk a bemeneti adatok is lehetnek olyanok, hogy összességében biztosan nem készíthető velük minden követelményt kielégítő beosztás.

Dolgozatomban ezekre a problémákra keresek megoldást. Kidolgoztam egy saját módszert és modellt, amelyekkel beosztástervezési feladatok változatos követelményeit lehet formalizálni, és a szabályok alapján csökkenteni az állapotteret. A lecsökkentett állapottér más heurisztikák és megoldóalgoritmusok alapjául szolgálhat. Továbbá a módszer segítségével az is detektálható, ha a bemeneti adatokban vagy követelményekben ellentmondás van, és nem készíthető beosztás.

1.2 A dolgozat felépítése

A következő fejezetben először is röviden ismertetem a problémakörhöz kapcsolódó szakirodalom eredményeit.

Ezután részletesen bemutatom a beosztástervezési problémák állapotterét csökkentő saját gráf-alapú modellem ötletét, felépítését és működését. Kifejtem a speciális elemeit, a rajta végezhető műveleteket, és ezek tulajdonságait. Példákkal illusztrálom az egyes szabályokat, és leírom a modellen megjelenő jelölések jelentését. A fejezetben végezetül további algoritmusokat is bemutatok, amelyek a gráfon használhatók abból a célból, hogy még tovább csökkentsék az állapotteret.

Ezt követően ismertetem a BME egy 100 fős heterogén hallgatói csoportjának záróvizsgabeosztás-készítésének a teljes problémáját. A modellemen formalizálom az összes szigorú és gyenge szabályt.

Végezetül bemutatom tesztprogramom eredményeit a konkrét záróvizsgabeosztás-készítési feladatnál.

2 Irodalomkutatás

A beosztástervezés egy hosszú ideje kutatott téma, mivel számos probléma megfogalmazható általa. Ebbe tartozik dolgozók munkahelyi beosztása, órarendek készítése, vagy akár erőforrások megfelelő allokálása is. A téma sokszínűségét mutatja, hogy számos összefoglaló cikk is készül folyamatosan a témában, jellemzően egy-egy témakörre specializálódva. Gyártásoptimalizálás kapcsán készült beosztások összefoglalásáról írt Chaudhry és Khan cikket [3], Pillay automatikus órarendkészítésről írt [4], míg a felhő alapú rendszerekről (mely szintén egy igen nagy irodalommal rendelkező résztéma) Vijindra értekezett [5]. Rendkívül sokféle követelmény fogalmazható meg [6], így még napjainkban is számos kihívást rejt a téma, mivel többségében bizonyítottan NP-teljes problémákról beszélhetünk [7].

A záróvizsgabeosztás-készítés a beosztástervezési feladatok egy speciális részfeladata. Célja az, hogy egy felsőoktatási intézmény megadott vizsgaidőszakára vonatkozóan elkészítsük a hallgatók és oktatók beosztását a megadott követelmények szerint. Az egyes felsőoktatási intézmények eltérő követelményeket támaszthatnak, és általában egyszerre rengeteg különböző szigorú és enyhe feltételt is figyelembe kell vennünk a beosztások készítése során.

Az órarendtípusú beosztástervezések problémáját három különböző altípusra bontják az irodalomban [8]. Ez a School timetabling, tehát az általános vagy középiskolai órarend készítése, a Course timetabling, vagyis a felsőoktatási kurzusok beosztása, és az Examination timetabling, vagyis a felsőoktatási vizsgabeosztás. Ezek közül az Examination timetabling áll a legközelebb a záróvizsga-beosztás problémájához, azonban mégis jelentős eltérések találhatóak az alapvető követelmények szintjén, mivel ennek az egyetemi hallgatók különálló vizsgáinak minél optimálisabb szétosztása a célja.

Mivel számos, teljesen különböző záróvizsgabeosztás-készítési feladat fogalmazható meg, így az azok megoldására használt algoritmusok jelentősen eltérhetnek egymástól, és más problémáknál nem használhatók fel. Az általam kutatott feladathoz Erdős tanulmányai állnak a legközelebb. A 2019-ben írt tanulmányában [9] bemutatta a jelen dolgozatban is ismertetett záróvizsgabeosztás-készítés egyszerűsített szabályait, és magyar módszert használt a probléma megoldásához. Egyszerűsített

szabályokkal dolgozott még a genetikus algoritmus [10], lineáris programozás [11], és neurális hálózat [12] alapú megoldásaiban. Az egyszerűsített szabályok keretein belül a hallgatók egy homogén csoportjának készítette el a beosztását, tehát az egy képzéshez tartozó hallgatóknak. Továbbá nem öt perces időintervallumokba osztotta be a vizsgákat, hanem rögzített időszeletekbe. Így ezek a módszerek nem használhatóak fel közvetlenül a teljes záróvizsgabeosztás-készítési problémához, pedig annál a feladatnál is már egy 100 fős hallgatói halmazra készíthető beosztások száma 10^{462} nagyságrendű volt. 2020-as tanulmányában [13] kifejti a lineáris programozás alapú algoritmus problémáit az öt perces időintervallumok és párhuzamos vizsgák kezelésekor. Az állapottér kezelhetetlen nagyságúra nőtt ebben az esetben, több heti futásidővel sem sikerült a szigorú követelményeket teljesítő beosztást készíteni az egyszerűsített modell ilyen módú bővítésével.

Varela és Soto 2002-es tanulmányukban [14] egy állapotteret csökkentő algoritmust mutatnak be a feladatütemezés problémájához. Azonban ez a beosztástervezés egy speciális részproblémája, a megoldásuk nem tudja kezelni a záróvizsgabeosztás-készítésben felmerülő sokféle követelményt, továbbá a gyenge szabályokat sem.

Az irodalomban használnak gráf-alapú megközelítéseket beosztástervezési problémák reprezentációs formájaként, mint például Cheng, Gen és Tsujimura (1996) [15], vagy Wang, Fan, Zhang és Wan (2014) [16]. Azonban ezeknél a gráf nem szolgál konkrét egyedi algoritmusok elkészítésére.

Dolgozatomban egy teljesen saját módszert dolgozok ki, amivel változatos követelmények modellezhetők, azon célból, hogy majd több beosztástervezési feladathoz is alkalmazható legyen. A megadott követelmények alapján az algoritmus redukálja az állapotteret, ami más algoritmusok és módszerek alapjául szolgálhat. Egy gráf-alapú modellt alkottam, amelyben az entitások közötti különböző kapcsolatokat is csúcsokként reprezentálom, és saját szabályokat definiálok, amelyek segítségével az állapottér folyamatosan csökkenthető.

3 A gráf-alapú modell

Az alábbi fejezetben mutatom be a beosztástervezési problémák állapotterének csökkentésére szolgáló saját modellezési módszeremet és algoritmusaimat.

3.1 Az ötlet

A beosztástervezési problémák során megadott követelményeknek eleget téve kell különböző entitásokat társítani egymással, és a különböző szabályos beosztások közül a lehető legoptimálisabbat megtalálni. A módszerem lényege, hogy hatékony módon nyilvántartsuk, mely összerendelések lehetségesek az addigi tudás alapján, és milyen állapotokat zárhatunk ki teljesen. A lehetséges összerendeléseken folyamatosan iterálhatunk, különböző algoritmusok segítségével egyre szűkíthetjük ezek terét, elméletileg akár addig is, amíg meg nem kapjuk az összes szabályos beosztást.

A záróvizsgabeosztási problémák hatalmas állapottere és NP-teljessége miatt az előbb említett módon legtöbb esetben túlságosan időigényes lenne teljes beosztás készítése, de már az is hasznos lehet más algoritmusok számára, ha sikerül lényegesen csökkenteni a lehetséges állapotok számát. Könnyedén kizárhatóak a teljesen ellentmondásos állapotok, vagy akár az is észrevehető, ha az általunk definiált követelmények és bemeneti adatok ellentmondásossága következtében nem is készíthető jó beosztás. A lecsökkent állapottér tetszőleges algoritmusok kiindulási pontjaként szolgálhat. Továbbá a módszer folyamatosan is alkalmazható lehet más algoritmusok futása közben, azok bizonyos lépései után megvizsgálva, hogy szabályos-e a kapott állapot, így végül bizonyos szigorú feltételeknek biztosan megfelelő beosztást készíthetünk.

3.1.1 A módszer működésének alapja

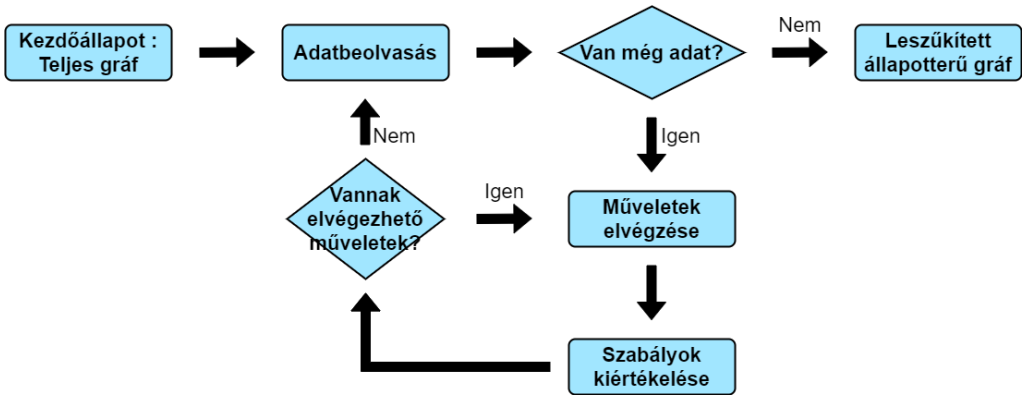
A beosztástervezési probléma során felmerülő lehetséges összerendeléseket megjeleníthetjük gráfon, minden entitáshoz egy csúcsot rendelve, és ha a jelenlegi tudás szerint megengedett azok társítása, akkor a nekik megfelelő csúcsokat éllel összekötve. Így gráfunkban a társítási lehetőségeket tároljuk el. A gráf alapértelmezett állapotában minden csúcs minden másik csúccsal össze van kötve, minden társítás megengedett, így a teljes állapottér jelenik meg. Ezen később algoritmusok segítségével módosítunk, a

megadott kritériumok alapján töröljük a nem megfelelő társításoknak megfelelő éleket, vagy akár csúcsokat is a gráfból. A cél minél hatékonyabb algoritmusok használata, amelyek a lehető legkevesebb vizsgálati lépés elvégzése után törlik a lehető legtöbb nem megfelelő társítást, és így csökkentik az állapotteret.

A kritériumok hatékony vizsgálatára ötlet, hogy a gráf csúcsaira egyesével támasszunk feltételeket. Ha valamilyen változás történik a gráfban, akkor lehetőség szerint csak a releváns csúcsoknál vizsgáljuk meg a művelet hatását, nem pedig az egész gráfnál. Így nagyon sok felesleges számítást kerülhetünk el.

Az egyes csúcsokat különböző típusokba sorolhatjuk, és a típusok szintjén követelményeket határozhatunk meg. Ilyen követelmény lehet például, hogy amennyiben egy adott típusba tartozó pontból nem fut egyetlen él sem valamely másik típus pontjába, akkor a pont törölhető. A csúcs törlése egy művelet, amely hatására újabb vizsgálatok történnek, amelyek szintén újabb műveleteket vonhatnak magukkal. Így lényegében egy művelet hatása végigterjedhet az egész gráfon, újabb és újabb műveleteket létrehozva, és ezek mindig csak lokális vizsgálatokat igényelnek.

A beosztástervezési feladatok állapotterét csökkentő módszerem alapvető működési elve a következő: A beosztástervezési problémában megjelenő entitásokat reprezentáló csúcsokból készítsünk egy teljes gráfot. A csúcsokat soroljuk típusokba, és a probléma követelményeit fogalmazzuk meg a csúcsok és típusok szintjén. Ezután a rendelkezésre álló bemeneti adatokat sorban vigyük be a gráfba, amit a gráfon végzett műveletekkel tehetünk meg. Az egyes műveletek hatására megtörténik a kritériumok vizsgálata, és ezek más újabb műveleteket vonhatnak maguk után. Miután megtörtént az összes adat bevitele, egy leszűkített állapotteret kapunk, amelyből az összes olyan lehetséges társítás törölve lesz, amely egyértelműen ellentmond az általunk definiált kritériumoknak.



1. ábra - Az állapotteret csökkentő algoritmus alapvető működése

Az adatok bevitele után kapott gráf további vizsgálatok és algoritmusok alapjaként szolgálhat. Megadhatunk olyan feltételeket, amelyek megmutatják, hogy a gráf jelenlegi állapota alapján készíthető-e beosztás, vagy pedig szabálytalan az állapota. Próbálgatásokat végezhetünk a gráfon, hogy mi történik bizonyos műveletek hatására. A próbálgatások után is automatikusan kiértékelődnek a követelmények, és ha szabálytalan állapotba kerülünk, akkor tudhatjuk, hogy nem lett volna szabad elvégezni a műveletet. Ezt a tudást felhasználva pedig további műveleteket végezhetünk el.

3.1.2 A modellezett szabályok tulajdonságai

Az algoritmus helyes működéséhez szükség van arra, hogy a gráf egyes csúcsaira olyan követelményeket támasszunk, amelyek egy tetszőleges műveletsorozat műveleteinek sorrendjétől függetlenül ugyanazt a végállapotot eredményezik a teljes sorozat lefutása után. Erre azért van szükség, hogy az adatok beolvasásának sorrendjétől függetlenül ugyanazt a leszűkített állapotteret kapjuk meg, továbbá azért, hogy biztosan minden követelmény ellenőrzésre kerüljön.

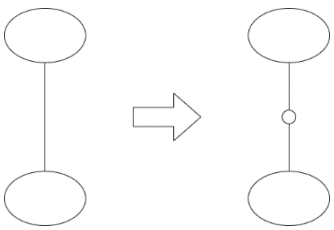
Ezen felül követelményeinket úgy is meghatározhatjuk, hogy ne konkrétan a műveletekhez, hanem csak a gráf aktuális állapotához kötődjenek. Így nincs is szükség arra, hogy teljes gráf legyen a kiindulóállapot, az adatok alapján elkészíthetünk egy alapból csökkentett állapotteret reprezentáló gráfot, amire kiértékelhetjük követelményeinket, amelyek hatására továbbra is újabb és újabb műveletek kerülhetnek elvégzésre. Ez nagyban gyorsítja az algoritmus futását, hiszen nem szükséges kezdetben a rengeteg élből és pontból álló teljes gráfon dolgozni, hanem rögtön rendelkezésre áll egy már csökkentett állapotteret reprezentáló gráf.

A gráf a lehetséges entitásokat és azok lehetséges társításait jeleníti meg. Így a legfontosabb műveletek a csúcsok és élek törlése. Ezek elvégzésével tarthatjuk nyilván, ha valamely összerendelések nem tehetőek meg, vagy valamely entitás nem használható fel a beosztáshoz. Ezenfelül viszont vannak olyan társítások és entitások, amelyekről tudjuk, hogy biztosan részei kell legyenek a kész beosztásnak. Ezért a modellben lehetőség van egyes csúcsok vagy élek kiválasztására. A kiválasztások is műveletek, és ezek hatására is történhet különböző szabályok vizsgálata a korábbiakban leírtaknak megfelelően. Egy teljesen elkészült beosztás gráfjában csakis kiválasztott élek és csúcsok maradnak.

Eddig a szigorú követelményeket vizsgáltuk, tehát az egyértelműen rossz állapotok kizárását. A gyenge követelmények azt adják meg, hogy több elfogadható beosztás közül melyik mennyire optimális. A modell képes lehet ezek kezelésére is. Minimális és maximális büntetőpontszám rendelhető az egyes csúcsokhoz, továbbá olyan szabályok, amelyek azt adják meg, hogy a gráf aktuális állapota alapján mennyivel növelhető a minimális büntetőpontszám, és mennyivel csökkenthető a maximális. A korábban leírt állapotteret szűkítő módszer tehát kiterjeszthető büntetőpontok kezeléséhez is. Így minden állapotnál becslést kaphatunk az aktuálisan elérhető minimális és maximális összes büntetőpontszámról. Ezek a büntetőpontszámok megadják, hogy az egyes állapotok mennyire optimálisak, aminek különböző heurisztikáknál lehet szerepe, továbbá kizárhatunk olyan állapotokat, amelyek biztosan nem megfelelőek.

3.1.3 Kapcsolatok reprezentálása a modellben

Az állapotteret csökkentő módszer hatékony működéséhez szükség van arra, hogy a műveletek hatására történő követelményvizsgálatok a lehető legkevesebb számítást vegyék igénybe. Két típusra és a köztük futó élekre könnyen tudunk ilyen feltételeket szabni, például az élek számosságát figyelhetjük. Ha azonban a beosztástervezési probléma követelményei között definiálva van olyan is, amely három vagy még több típust érint, akkor új ötletre van szükségünk. A cél az, hogy a műveletek hatására történő kritériumvizsgálatok mind konstans számításigényűek legyenek. Ez megvalósítható úgy, hogy a gráfunkban magukat a kapcsolatokat is csúcsokkal reprezentáljuk. Tehát két csúcs között futó élt helyettesíthetünk egy új csúccsal, amelyet összekötjük a másik két csúccsal. Ennek az új csúcsnak is van típusa, és erre is megadhatóak követelmények.



2. ábra - Két csúcs közötti kapcsolat reprezentálása csúccsal

Több ilyen kapcsolatot reprezentáló csúcs is kapcsolatban állhat egymással, és ezt az újabb kapcsolatot is reprezentálhatja egy csúcs. Így tehát tetszőleges kapcsolatokhoz megadhatók csúcsok a modellben, és ezekre a csúcsokra alkalmazható

minden eddig taglalt módszer, továbbá könnyen definiálhatók segítségükkel konstans számításigényű vizsgálattal kiértékelhető követelmények. A módszer hátránya, hogy több csúcsot és élt kell használni a gráfban. Azonban a követelmények vizsgálata sokkal gyorsabb lesz, továbbá az állapotokról több információt is számon tudunk tartani, így lehetőleg még jobban csökkentve az állapotteret.

3.2 A modell felépítése, műveletek

3.2.1 A modell alapvető elemei

A modell alapja egy irányítatlan, egyszerű gráf. A gráf csúcsai különböző entitásokat reprezentálnak. A csúcsok lehetnek kiválasztottak, vagy kiválasztatlanok. A kiválasztott csúcsokra mindenképp szükség van a beosztás szempontjából, a kiválasztatlanokról pedig még nincs eldöntve szükségességük.

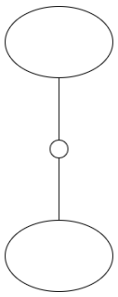
A csúcsok akkor vannak összekötve más csúcsokkal, ha lehetséges, hogy valamilyen kapcsolatban állnak. Tetszőleges kapcsolatokat modellezhetünk ilyen módon, például különböző társítási lehetőségeket, vagy akár azt is, ha megadott típusokba sorolhatóak bizonyos entitások. Az élek lehetnek valamelyik oldalukról, vagy teljesen kiválasztottak, de nem lehetnek teljesen kiválasztatlanok, aminek miértjére a későbbiekben térek vissza. Ha egy él ki van választva egy csúcsból, az azt jelenti, hogy amennyiben a csúcs bekerül a kész beosztásba, hozzá biztosan társítani kell az él túloldalán levő csúcsot. Ha egy él nincsen kiválasztva valamely csúcsból, az azt jelenti, hogy még nincs eldöntve a társítása az él túloldalán levő csúccsal.

A csúcsokat különböző típusokba soroljuk, így a típusuk szerint halmazokba lehet csoportosítani őket. A típusaik alapján tudjuk megadni a modellben az egyes pontok kapcsolatait, és rájuk vonatkozó szabályokat. Minden csúcs pontosan egy típushoz tartozik. Az éleknek is van típusa, ezt az határozza meg, hogy milyen típusú pontok között fut. Egy él futhat két azonos típusú pont között is.

3.2.1.1 Speciális csúcsok

Ahhoz, hogy a későbbiekben könnyedén definiálhassunk szabályokat több típusra vonatkozóan, az ötletem az, hogy a köztük lévő kapcsolatokat is a gráf csúcsaiként modellezzük. Ha két csúcs közötti kapcsolatot szeretnénk megadni, akkor a köztük futó élt helyettesítsük egy új csúccsal, amelyet kössük össze a másik két

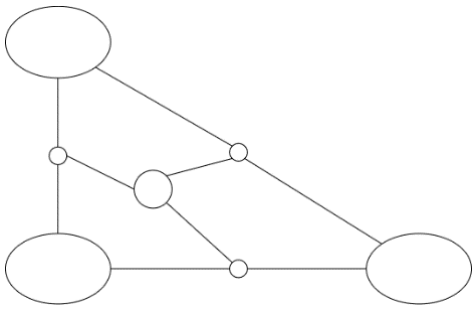
csúccsal. Az ilyen csúcsokra a továbbiakban élcsúcsként, élpontként vagy élkapcsolatként fogok hivatkozni.



3. ábra - Élcsúcs

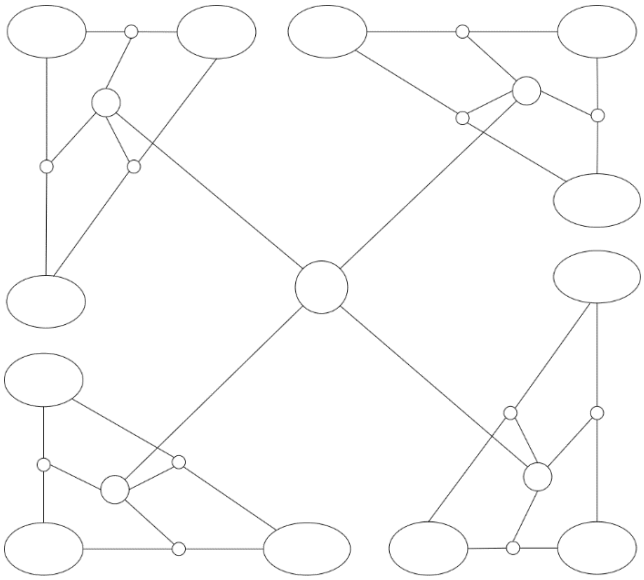
Az élcsúcsból az eredeti él felbontásával kapott mindkét él ki van választva. Hiszen egy kapcsolat csak akkor értelmezhető, ha társítva van azokhoz az entitásokhoz, amelyekre értelmezett.

Három csúcs közötti kapcsolatot is modellezhetünk ponttal. Ezt úgy tehetjük meg, hogy minden lehetséges pontpár közötti kapcsolatot egy-egy élponttal modellezzük, felveszünk egy új csúcsot, és ezt mindhárom élponttal összekötjük. Az ilyen kapcsolatot reprezentáló csúcs neve háromszögcsúcs, háromszögpont vagy háromszögkapcsolat. A háromszögpont mindhárom élpontba tartó éle ki van választva, ha bármely két pont közötti kapcsolat megszűnik, úgy a hármuk közötti kapcsolat is.



4. ábra - Háromszögkapcsolat

Négy csúcs közötti kapcsolatot tetraéderkapcsolattal adhatunk meg. Ehhez a négy csúcs közül az összes lehetséges hármas kapcsolatát meg kell adnunk egy háromszögkapcsolattal. A tetraéderpontból mind a négy háromszögpontba tartó él ki van választva.

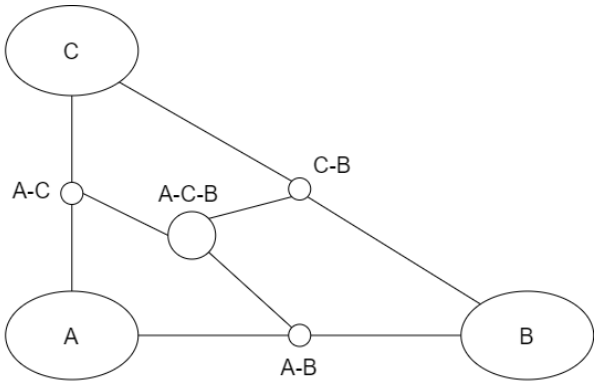


5. ábra – Tetraéderkapcsolat

Az előbbiekhez hasonlóan elméletileg tetszőleges számú csúcs kapcsolatát is modellezhetnénk pontok segítségével. Azonban ez eléggé átláthatatlan lenne, ráadásul nagyon sok új pont és él felvételét igényelné. Ezért olyan esetekben, amikor több mint négy típusra szeretnénk követelményeket megfogalmazni, összevonhatunk típusokat egymással. Két típus összevonása azt jelenti, hogy az egyes típusokba tartozó pontok által alkotott halmazok Descartes-szorzatát vesszük, és ezen új halmazt használjuk modellünkben típusként az eredeti két típus helyett. Egy A és B típus összevonásával keletkezett típusra az „A x B” nevet használom a dolgozatban. Megfelelő számú típusösszevonással elérhető, hogy legfeljebb csak tetraéderkapcsolatokat használjunk a modellünkben. Minden esetben külön vizsgálatot érdemel, hogy mely típusokat érdemes összevonni. Ez attól függ, hogy milyen követelményeket szeretnénk modellezni és hogyan tudjuk minimalizálni a pontok és élek számát.

A modellben minden kapcsolatot reprezentáló élt cseréljünk egy kapcsolatpontra az eddig leírtak szerint. Így az élek kizárólag egy entitást reprezentáló pont és egy kapcsolatpont, vagy pedig két kapcsolatpont között futhatnak. Ez azt is vonja maga után, hogy az élek nem lehetnek teljesen kiválasztatlanok, mindig legalább az összetettebb kapcsolatot reprezentáló pont felől kiválasztottak.

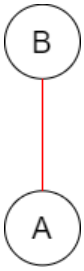
Az A, B, … típusok kapcsolatpontjának típusára az „A-B-…” névvel hivatkozhatunk, ahol a felsorolás sorrendje nem releváns. Ezen felül az A és B típus közötti élpont esetén az „A<->B” elnevezést is használom a dolgozatban.



6. ábra – Példa kapcsolattípusok elnevezésére

3.2.1.2 Kiválasztottság és büntetőpontszám

Ha tudjuk, hogy egy típusból rendelkezésre áll az összes kiválasztott csúcs, akkor ezt piros körvonal jelzi a modellen. Ha egy éltípus minden eleme teljesen kiválasztott, akkor ezt piros színű él jelzi. Ezeket kiválasztott típusoknak nevezem a dolgozatban. Minden éltípusnak egyértelműen ki van választva az egyik fele, így azzal a kiválasztottsággal sosem kell foglalkoznunk. Ha egy A típusból ki van választva a B típus felé futó éltípus, azt úgy rövidítem, hogy „Az A típusból ki van választva a B típus”. Erre látható példa a 7. ábrán. Típusok helyett a konkrét csúcsok és élek modellezésénél is használhatjuk ugyanezeket a jelöléseket a kiválasztott csúcsok/élek megjelenítéséhez.

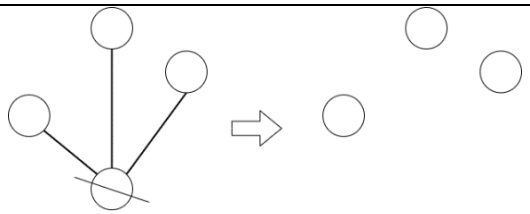
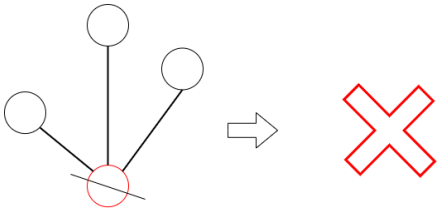
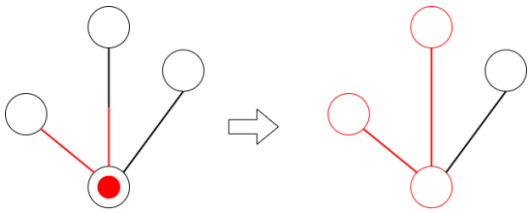
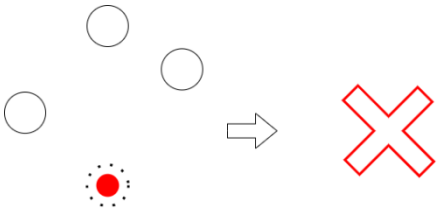
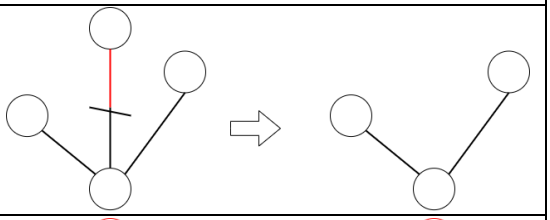
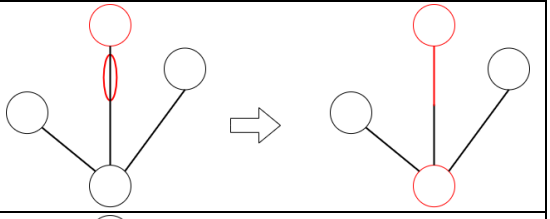
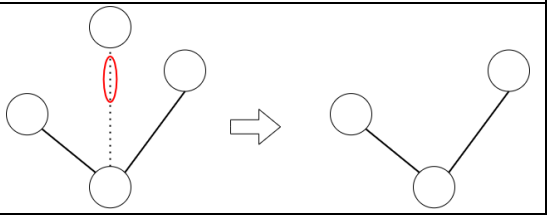


7. ábra – Kiválasztott éltípus ábrázolása a modellen

Az egyes csúcsokhoz minimális és maximális büntetőpontszámokat határozhatunk meg. Az algoritmus a futása közben megadott szabályok alapján növelheti a minimális büntetőpontszámot, és csökkentheti a maximálisat, így egyre jobb becslést adva a tényleges büntetőpontszámot illetően. A csúcsok büntetőpontszámából kiszámolható az egész gráf minimum és maximum büntetőpontszáma is.

3.2.2 Műveletek, szabályok

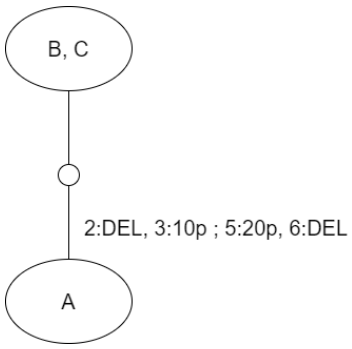
A gráfon négy művelet elvégzése lehetséges: csúcs/él törlése/kiválasztása. Ezekre minden esetben vonatkozik néhány alapvető szabály:

Ha egy csúcsot törlünk, akkor minden belőle futó él is törlendő.	
Ha egy kiválasztott csúcsot törlünk, az ellentmondásos állapotot eredményez, ami azt jelenti, hogy a megadott követelmények és adatok alapján nem készíthető beosztás.	
Ha egy csúcsot kiválasztunk, akkor minden kiválasztott élének túloldalán található csúcs is kiválasztható, továbbá azokból a csúcsokból is kiválasztható az él.	
Ha egy olyan csúcsot választanánk ki, amely már törölve lett, akkor ellentmondásos állapotot kapunk, tehát nem készíthető beosztás.	
Ha egy csúcsból kiválasztott élet törlünk, akkor törlendő a csúcs is.	
Ha egy kiválasztott csúcsból választunk ki egy élt, akkor az él túloldalán található csúcs is kiválasztható.	
Ha egy csúcsból egy olyan élet választanánk ki, amely már törölve lett, akkor törlendő a csúcs.	

A modellezni kívánt probléma követelményeit a csúcsok szintjén fogalmazhatjuk meg. A modellben tetszőleges pontok halmazára megadhatunk szabályokat. Ez a halmaz lehet bizonyos típus, típusok uniója, vagy csak néhány tetszőleges pont által alkotott halmaz is. A megfogalmazott követelmények a megadott halmaz minden egyes elemére külön vonatkoznak. Magában a szabályban csak típusokra vagy ezek uniójára hivatkozhatunk. Ha a hivatkozni kívánt csúcsokra még nem létezik típus, akkor szükség esetén létrehozható új. Kétféle szabály alkalmazható a modellben, ezeket a megfelelő éltípusokra írva adhatjuk meg.

3.2.2.1 Élek számosságára vonatkozó szabály

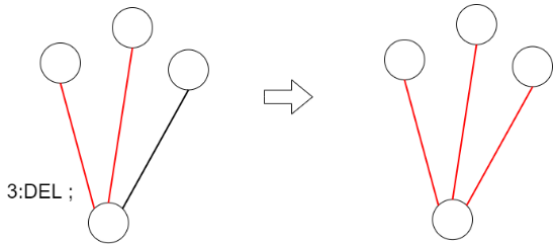
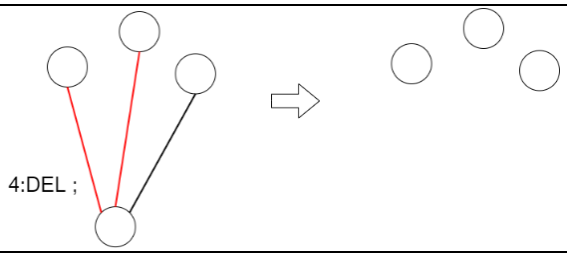
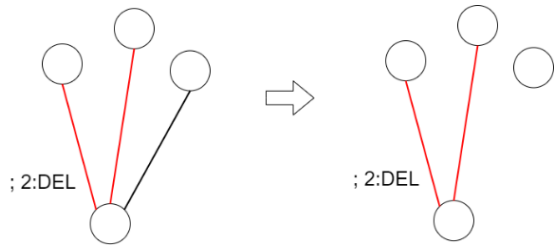
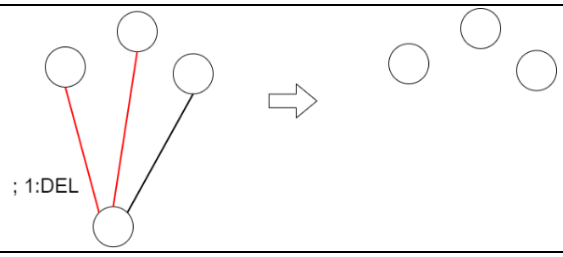
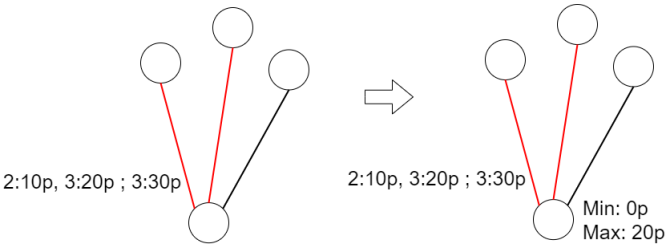
Az első szabálytípus egy csúcsból kimenő, megadott típusok uniójának csúcsai felé tartó élek számára vonatkozik. Megadható, hogy minimum és maximum mennyi ebbe a halmazba tartozó él futhat ki egy pontból. Továbbá megadható, hogy bizonyos éldarabszám alatt vagy felett mennyi büntetőpontot kell adni. A szabályt a következő formában írhatjuk le: „{Minimum élszámra vonatkozó követelmények} ; {Maximum élszámra vonatkozó követelmények}”. Ha nincsenek minimum vagy maximum élszámra vonatkozó követelmények, akkor az a rész üresen hagyandó. Az egyes részekben felsorolhatóak a vizsgálandó élszámok, és megadható, hogy kevesebb/több élszám esetén törölni kell a pontot, vagy csak a megadott büntetőpontszámot kell hozzáadni a meglévőhöz. „{Élszám}:DEL” a jelölés, ha törlést kell végezni a szabály megsértése esetén és „{Élszám}:{Pontszám}p" jelöli, ha a megadott büntetőpontszámot vonja magával a szabály megsértése.



8. ábra - Példa az élek számosságára vonatkozó szabályra

A 8. ábra egy élek számosságára vonatkozó szabályt mutat be. Az A típus minden pontjából az A-B és A-C típusok felé összesen legalább 2, legfeljebb 6 élnek kell futnia. 3-nál kevesebb él esetén 10 büntetőpontot eredményez a szabály, 5-nél több esetén pedig 20-at.

Az élek számosságára vonatkozó szabály alapján a következő műveletek végezhetőek el:

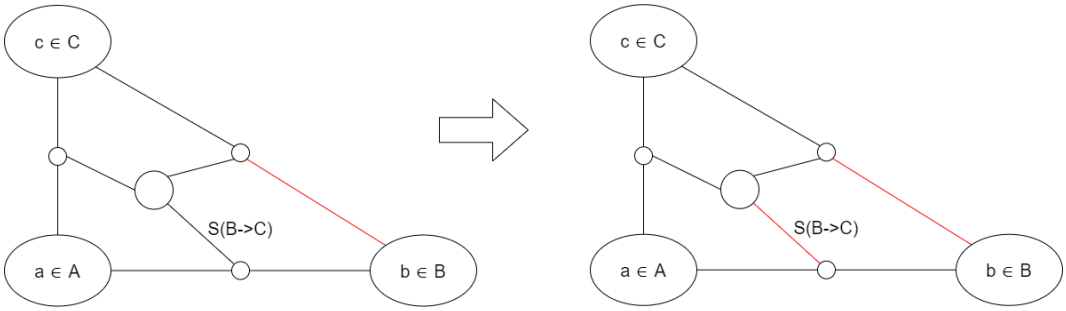
Ha annyi él fut ki a csúcsból a típusok egy halmazába, amennyi minimális feltételként meg van adva, akkor ezek mind kiválaszthatóak.	
Ha a minimálisnál kevesebb, akkor törlendő a csúcs.	
Ha annyi él ki van választva a típusok egy halmaza felé egy csúcsból, amennyi maximális feltételként meg van adva, akkor minden további, a halmazba tartó kiválasztatlan él törölhető.	
Ha a maximálisnál több, akkor a csúcs törlendő.	
A csúcsból a megadott típusok halmazába tartó összes élszám és a kiválasztott élek száma alapján meghatározható a szabály által adott minimális és maximális büntetőpontszám.	

Egy csúcs minimum/maximum büntetőpontszáma a csúcs alap minimum/maximum büntetőpontszáma és a csúcsra vonatkozó egyes szabályok által keletkeztetett minimum/maximum büntetőpontszámainak összegeként áll elő. Az egész

gráf minimális büntetőpontszáma a kiválasztott csúcsok minimális büntetőpontszámainak összege, a maximális pedig az összes csúcs maximális büntetőpontszámainak összege.

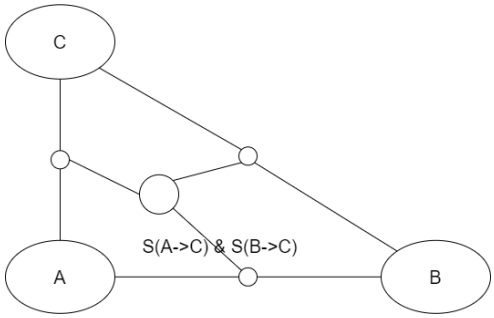
3.2.2.2 Kiválasztást kényszerítő szabály

A modellben használható második szabálytípus élek kiválasztásának szükségességét mondja ki megadott más élek kiválasztottsága alapján. Egy kapcsolatpont összetettebb kapcsolat felé futó éleire tehetünk követelményeket a kapcsolatponttal összeköttetésben álló entitást vagy egyszerűbb kapcsolatot reprezentáló pontból futó élek alapján. A szabály jelölése „S({Típusok uniójának nevének a rövidítése}->{Típusok uniójának nevének a rövidítése}”. Az A típusból az A-B kapcsolat felé futó éltípusára megadott S(B->C) szabály jelentése: Ha egy b $\in$ B pontból ki van választva egy b-c (c $\in$ C) pont, akkor minden b-a (a $\in$ A) pontból kiválasztandó a-b-c, amennyiben létezik ilyen, és amennyiben nem létezik, úgy az a-b pont törlendő.



9. ábra – Példa élek kiválasztását kényszerítő szabály működésére

Egy kapcsolattípusból induló éltípusra több típus vonatkozásában is megfogalmazhatunk kiválasztást kényszerítő szabályt, ezeket „&” jellel összefűzve. Ilyenkor egy adott él, amire a követelmény vonatkozik akkor választandó ki, ha az összes megadott kiválasztást kényszerítő szabály miatt egyszerre kiválasztandó.

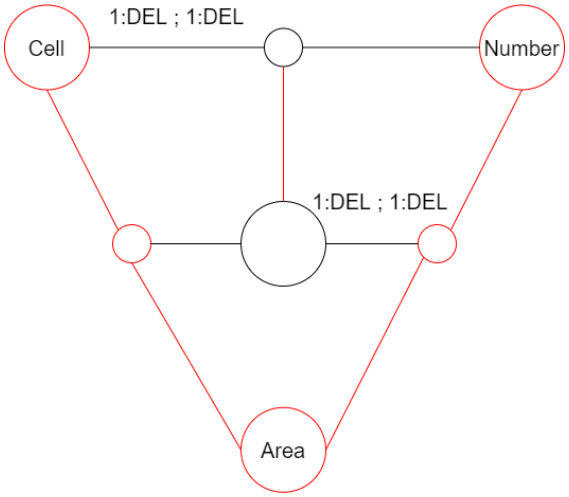


10. ábra - Élek kiválasztását kényszerítő szabály jelölése több típus vonatkozásában

3.2.2.3 A szabályok tulajdonságai

Egy éltípusra egyszerre több élek számosságára vonatkozó és kiválasztást kényszerítő szabály is megadható a modellben, ezeket felsorolhatjuk a megfelelő éltípusnál. Az összes szabály fontos tulajdonsága, hogy műveletek elvégzése után pontosan lehet tudni, melyik csúcsoknál mely követelményeket kell megvizsgálni. Továbbá a kezdeti adatokon is elvégezhető a kiértékelésük, és nem függ az eredményük a végzett műveletek sorrendjétől.

A bemutatott szabályokkal könnyen leírhatjuk sokféle összerendelési probléma követelményeit. Példaként a 11. ábra tetszőleges méretű sudoku szabályainak modellezését mutatja be. Nem tartalmazza a modell, hogy pontosan milyen számok és cellák vannak és ezek milyen részterületeken helyezkednek el, ezeket az információkat a bementi adatokból kell beolvasni. A módszer képes a modell alapján lecsökkenteni az állapotteret, a később ismertetett algoritmusokkal pedig akár teljesen megoldani is a sudokut.

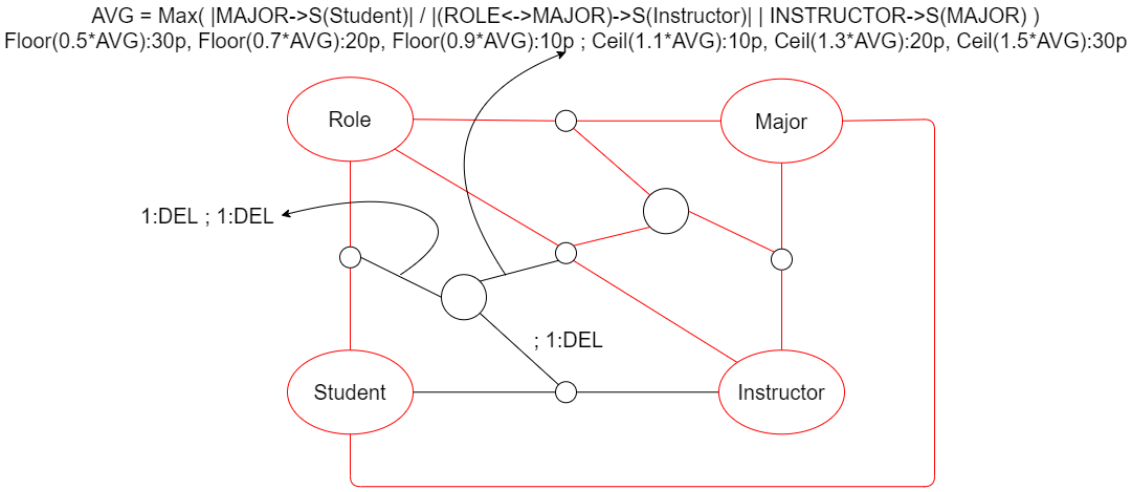


11. ábra - Sudoku követelményeinek modellezése

3.2.3 Jelölések bemeneti adatok alapján felvett szabályokhoz

Az egyes csúcsok és típusok szintjén megfogalmazott követelményeket meghatározhatjuk a bemeneti adatok alapján is. Ehhez modellünkben felhasználhatunk az eddig leírtaktól eltérő jelöléseket, ha ezek az új jelölések leképezhetőek a korábbi szabályokra. Minden ilyen jelölés azt határozza meg, hogy milyen korábban bemutatott szabályokat vegyünk fel az egyes pontokhoz.

A szabályok felvételének modellezéséhez használhatunk aritmetikai és logikai műveleteket, matematikai függvényeket, különböző halmazokra és ezek elemszámára utaló jelöléseket. A dolgozatomban található modellrészleteken egy típus nevét nagybetűvel kezdődő szóval jelölöm, ennek egy elemét pedig a típus nevének csupa nagybetűs változatával. Az „a->E(B)” jelölés az összes „a” pontból induló, B típusba tartó élet jelöli, az „a->S(B)” pedig az összes ilyen kiválasztott élet. A jelölésekből elhagyható a csúcs- és/vagy típusnév, ha egyértelmű, mire utal a szabály. Minden jelölés a bemeneti adatok alapján létrehozott gráf kezdeti állapotán van értelmezve.



12. ábra - Példa bementi adatok alapján felvett szabályokra

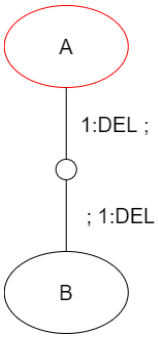
3.3 További algoritmusok

Az adatok bevitele, és a hozzá kapcsolódó műveletek elvégzése után kapott lecsökkentett állapotteret reprezentáló gráfon további algoritmusokat végezhetünk. Meghatározhatunk algoritmusokat elvégezhető műveletek keresésére, állapot szabályosságának eldöntésére, büntetőpontszám pontosabb becslésére, vagy akár a teljes beosztástervezési feladat elvégzésére heurisztikák segítségével. Az adott algoritmus számításigényétől függ, hogy milyen gyakran érdemes végezni, és milyen helyzetekben.

3.3.1 Következményszabályok

A modellben felvett szabályok és kiválasztottságok alapján sok esetben azonnal megfogalmazhatók további szabályok, sőt akár folyamatosan is változtathatjuk ezeket rekurzívan a gráfban, a műveletek végzéséhez hasonló módon.

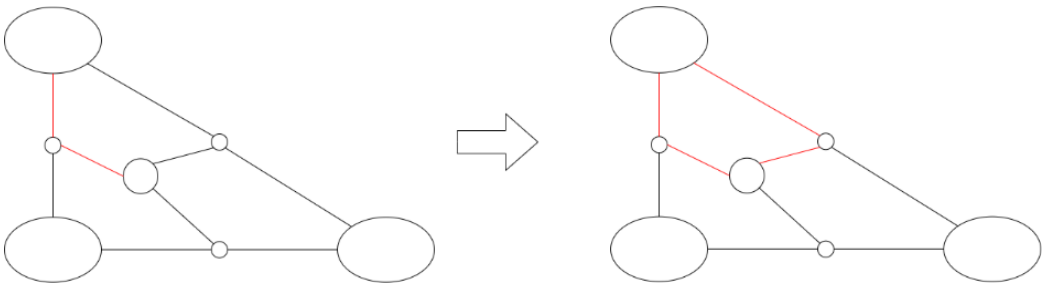
Felvehetünk úgynevezett következményszabályokat is. Ezek új szabálytípusok, amelyek az eddigi szabályok és kiválasztottságok alapján állnak elő és kiértékelésükhöz valamilyen egyéb algoritmusra van szükség.



13. ábra - Példa következményszabályra: A-t lefedő párosítás

A 13. ábra egy olyan modellrészletet mutat be, ahol a szabályok alapján következményszabályt alkalmazhatunk. Látható, hogy az A típus mindegyik pontjából legalább egy A-B pont választandó, és a B típus pontjaiból legfeljebb egy. Ez azt jelenti, hogy kell tudnunk készíteni A-t lefedő párosítást A és B között, különben valamelyik A pontnál nem teljesülne a megadott feltétel, és így az törlendő lenne. De mivel az A kiválasztott, nem törölhetjük egyetlen pontját sem, mert akkor szabálytalan állapotot kapnánk. Tehát az ábrán bemutatott szabályok esetén a következő következményszabály fogalmazható meg: Csak akkor ellentmondásmentes a gráf, ha létezik A-t lefedő párosítás A és B között. Ennek a következményszabálynak a kiértékelésére használhatjuk például a magyar módszert [17], amely futása polinomiális lépésszámú, folyamatosan számontartható, és így minden gráfon végzett művelet után elvégezhető.

Számos olyan következményszabály is definiálható, amely azt adja meg, hogy kiválasztott csúcsok vagy élek bizonyos láncolata miatt mely másik éleket vagy csúcsokat lehet kiválasztani. Erre mutat példát a 14. ábra.



14. ábra - Példa kiválasztási szabályra

Az összes kiválasztási szabály megfogalmazható a következőképpen: Ha egy „a” csúcs törlése miatt törlendő egy olyan „b” csúcs, amely össze van kötve az „a” csúccsal, akkor „b” csúcsból kiválasztható az „a” felé futó él. Így nem szükséges egyesével meghatározni a kiválasztási szabályokat és algoritmust implementálnunk hozzájuk, a kívánt működést elérhetjük a következőkben bemutatott próbálgatásokkal is.

3.3.2 Próbálgatások

A gráf lehetőséget biztosít arra, hogy kipróbáljunk bizonyos műveleteket, és megvizsgáljuk, ezek hogyan változtatják az állapotteret. Ha például ellentmondásos állapotot kapunk, tehát egy kiválasztott pont törlődik, akkor tudhatjuk, hogy a kipróbálás műveletét nem szabad elvégezni a gráfon. Ezen kívül még más információkat is leszűrhetünk a próbálgatások segítségével, és ezek alapján műveleteket végezhetünk a helyreállított eredeti állapoton.

Egy kipróbálás egy kiválasztatlan csúcs kiválasztását vagy törlését jelenti. Egy kész beosztásnál csak kiválasztott pontok maradhatnak, ezért minden kiválasztatlan pont vagy kiválasztásra, vagy törlésre kell, hogy kerüljön. Ennek segítségével a következő műveleteket tudjuk meghatározni egy kipróbálással:

- Ha a kipróbálás ellentmondásos állapothoz vezet, akkor a kipróbált művelettel ellentétes művelet elvégezhető az eredeti állapoton.
- Ha mindkét ellentétes művelet elvégzése ellentmondásos állapothoz vezet, akkor az eredeti állapot is ellentmondásos.
- Ha mindkét ellentétes művelet elvégzése bizonyos más műveleteket von maga után, akkor ezek elvégezhetőek az eredeti állapoton.
- Ha a kipróbált művelet törlés volt, és ez olyan csúcsok törlését vonta magával, amelyekből fut él a kipróbálás során eredetileg törölt csúcsba, akkor az eredeti állapotban ezekből a csúcsokból mind kiválasztható a kipróbálás során törölt csúcsba futó él.

Kipróbálások segítségével pontosabb becsléseket is kaphatunk a büntetőpontszámokról, továbbá az egyes csúcsokból futó élek minimum és maximum számosságát is változtatni lehet.

3.3.2.1 Faépítés állapotokkal és átmenetekkel

A gráfon kipróbálások sorozatát is végezhetjük, ilyenkor mindegyik kipróbálás után kiértékelődnek a követelmények, és tovább szűkül az állapottér. A kipróbálások után kapott állapotokat eltárolhatjuk valamilyen módon. Az állapotokat reprezentáló csúcsokból építhetünk egy fát, ahol a gyökér az eredeti állapot, és egy csúcs gyerekei az állapotból kipróbálásokkal kapott állapotok. Ahhoz, hogy megkapjunk egy állapotot, elég tudnunk, hogy a szülőjében pontosan milyen műveleteket kell végeznünk. Ezekbe a műveletekbe maga a kipróbálás és minden az által okozott, követelmények miatt elvégzett művelet is beleértendő. Így tehát, ha szeretnénk kipróbálás után megőrizni egy állapotot, elég azt elmenteni, milyen műveletek végzésével kapható meg. Így bármikor megkaphatjuk egy állapotból annak gyerekét, vagy pedig a műveletek visszacsinálásával annak szülőjét.

A kipróbálás során kapott állapotokat folyamatosan fent is tarthatjuk, ha az ősállapotokon végzett műveletet végrehajtjuk rajtuk. Ha valamelyik állapot ellentmondásos lesz, akkor annak szülőjén elvégezhető annak a kipróbálásnak az ellentétes művelete, amellyel az állapotot kaptuk. A kipróbált állapotok megtartása és vizsgálata számos módszer és heurisztika alapjaként szolgálhat. Ilyen módszer lehet például az ekvivalens csúcsok kezelése, a csúcsok összevonásának utánzása vagy a büntetőpontokra tett becslések. Például, ha kapunk egy kész beosztást, amelynek büntetőpontszáma x, akkor minden olyan állapotot ellentmondásosnak tekinthetünk, amelynek a minimum büntetőpontszáma nagyobb, mint x, hiszen ezek a beosztások biztosan rosszabbak lennének a kapottnál. Dolgozatom keretén belül ezekkel a módszerekkel és heurisztikákkal nem foglalkozok részletesebben.

Ha a gráf minden kiválasztatlan csúcsára az összes elvégezhető műveletet kipróbáljuk, úgy egy mélységű kipróbálást végeztünk. Több mélységű próbálgatás is lehetséges, ha az egyes kipróbálások után kapott állapotokban is kipróbálunk minden kiválasztatlan csúcsot. A különböző mélységű kipróbálásokat egymás után többször is érdemes lehet elvégezni, amíg tudunk új műveleteket meghatározni a segítségükkel.

4 A probléma formalizálása a modell segítségével

A Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar szóbeli záróvizsgájának leírását az egyetemi Tanulmányi és Vizsgaszabályzat [1], valamint annak kari kiegészítése [2] tartalmazza. Erdős 2018-ban írt tanulmányában [18] továbbá a GitHub oldalán [19] részletesen kifejti ennek a záróvizsga-beosztásnak az egyszerűsített követelményeit, és büntetőpontszámokat rendel az egyes gyenge feltételekhez. A dolgozatomban ezeket a büntetőpontszámokat és követelményeket, valamint a gyakorlatban használt metodikákkal is kiegészült szabályokat használom. Továbbá a teljes záróvizsgabeosztás-készítés problémájához még hozzá tartozik a párhuzamos szekciók, heterogén hallgatói csoportok és a nem optimális oktató-tantárgy párosítások kezelése is. Ezeken felül a modell elvárt működéséhez fel kellett vennem további segédszabályokat, többek között párhuzamos szekciók helyes kezeléséhez, szünetek beosztásához, vagy szekciók folytonosságának biztosításához. Az összes szabályt formalizáltam a modellem segítségével, ezt mutatom be a jelen fejezetben.

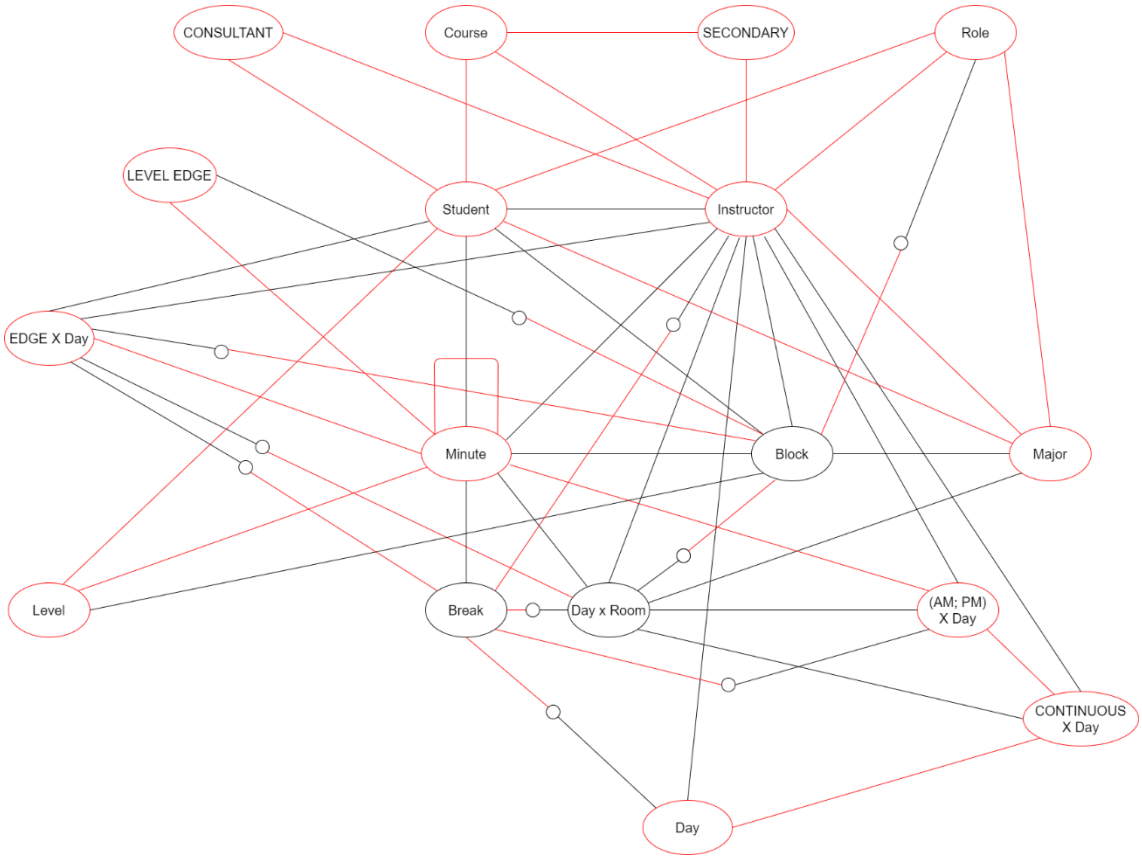
4.1 A modell típusai

A probléma modellezéséhez először is szükséges a megfelelő típusok felvétele, amelyek közti kapcsolatokkal adhatóak meg a különböző szigorú és enyhe követelmények. A típusok egy része értelemszerűen következik a leírásból, azokon felül viszont sok esetben szükség volt sajátok bevezetésére, amelyek segítségével összetettebb szabályok adhatók meg. Day (nap) típussal való összevonás történt több típusnál is. Erre azért volt szükség, hogy a bonyolultabb szabályok modellezésénél se kelljen tetraéderkapcsolatnál összetettebb kapcsolatokat használni. A modellben használt típusok a következők:

- Student: A hallgatókat és azok vizsgaalkalmát egyszerre reprezentáló csúcsok halmaza.
- Day: A záróvizsgabeosztás szempontjából releváns napok halmaza.

- Day x Room: A napok és az összes adott napon elérhető terem párjainak halmaza. Ezek egy szekciót reprezentálnak, ahol záróvizsgáztatás történhet.
- Block: A típus egy adott napon, egy adott teremben zajló, folyamatos vizsgáztatási szekciót reprezentál. A Day x Room minden eleméhez két Block típusú csúcs tartozik a modell kezdeti állapotában. Ezek a csúcsok a délelőtti és délutáni vizsgáztatási szekcióknak felelnek meg.
- Instructor: A záróvizsgákon tetszőleges szerepkörben részt vevő tanárok halmaza.
- Course: A záróvizsgákon választható tantárgyak halmaza.
- SECONDARY: Egy egyelemű halmaz, elemének neve azonos a típus nevével. Azon vizsgáztató-tantárgy kapcsolatok azonosítására szolgál, amelyek lehetőség szerint kerülendőek.
- Role: A tanárok vizsgákhoz, és blokkokhoz köthető szerepeinek halmaza. A halmaz elemei a President, a Secretary és a Member csúcsok, tehát rendre az elnök, a titkár és a belső tag szerepét reprezentáló csúcsok.
- CONSULTANT: Egyelemű halmaz, egyetlen elemének neve azonos a típus nevével. Vizsgáztató-hallgató közötti konzulensi kapcsolat megadására szolgál.
- Minute: 5 perces időintervallumokat reprezentáló csúcsok halmaza. A napok lehetséges legkorábbi vizsgakezdési időpontjától a legkésőbbi vizsgavégzési időpontjáig terjedő szakaszának 5 perces részekre bontásával keletkeznek az elemei.
- Major: A záróvizsgákon előforduló képzések halmaza.
- Level: A záróvizsgákon előforduló képzési szintek halmaza.
- Break: Szüneteket reprezentáló csúcsok halmaza. Szünete minden nap lehet tanárnak és teremnek is, így alapesetben minden Day és Instructor által alkotott pároshoz, valamint minden Day x Room eleméhez is tartozik egy csúcsa.

- CONTINUOUS X Day: Ezen csúcsok segítségével adható meg, ha bizonyos napokon nem tartozik szünet egy olyan entitáshoz, amelyhez tartozhatna.
- EDGE X Day: Ezen csúcsok segítségével adhatók meg olyan kapcsolatok valamely időtartományt reprezentáló csúcsok között, amely kapcsolatok azt jelentik, hogy a kisebb tartomány a nagyobbnak valamelyik szélén helyezkedik el egy adott napon.
- (AM; PM) X Day: Napszakokat reprezentáló csúcsok minden naphoz külön.
- LEVEL EDGE: Egyelemű halmaz, egyetlen elemének neve megegyezik a típus nevével. Blokk-perc közötti olyan kapcsolat megadására szolgál, amely azt reprezentálja, hogy képzésiszint-váltás történik a blokk azon percén.



15. ábra - A modell típusai, és azok egyszerűbb összeköttetései

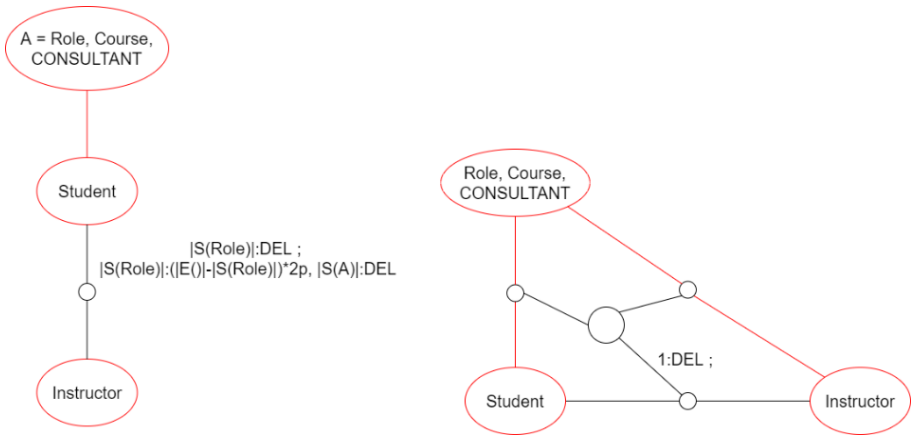
A 15. ábrán láthatók a modellben használt típusok, és azok egyszerűbb kapcsolatai. Minden csúcs és él a bemeneti adatok alapján kerül létrehozásra, a piros szín pedig azt jelenti, hogy az összes olyan csúcs- vagy éltípusba tartozó entitás kiválasztottsága is eldönthető az adatok alapján. Az egyes típusok között akkor van kapcsolat, ha ennek szerepe van a záróvizsga-beosztás szempontjából.

4.2 Követelmények

A típusok definiálása után következhet a szabályok megadása a modellben. A szigorú és gyenge követelmények egyszerre jelennek meg, a gyenge feltételek büntetőpontok formájában, a szigorú feltételek pedig kötelező követelményekként. Ahol a büntetőpontok leírásában [19] ezres nagyságrendű büntetőpont szerepelt, azt szigorú követelménynek tekinthetjük.

4.2.1 Hallgatók és oktatók kapcsolatai

Az első követelmények a hallgatók és oktatók kapcsolatára vonatkoznak. Egy hallgató vizsgáján akkor vesz részt oktató, ha ott legalább egy szerepet betölt. Egy oktató több szerepet is betölthet, azonban az elnöki, titkári és belső tagi szerep nem vonhatók össze egymással. A megengedett összevonások ideálisak is, így a minimálisnál több oktató részvételéért fejenként 2 büntetőpontszám számítandó egy vizsgaalkalmon.

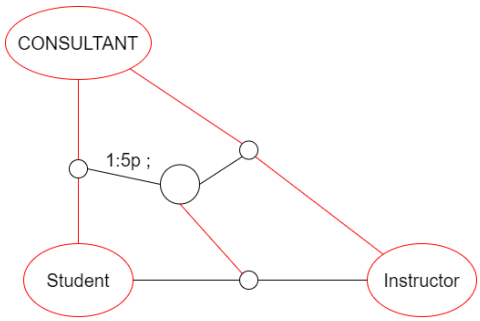


16. ábra - Hallgatók és oktatók kapcsolatára vonatkozó általános követelmények

A 16. ábrán láthatók a hallgatók és oktatók kapcsolatára vonatkozó feltételek. A Role ∪ Course ∪ CONSULTANT halmaz egy eleme akkor áll kapcsolatban a Student egy elemével, ha a hallgató vizsgáztatásánál releváns az adott szerepkör betöltésének vizsgálata, az Instructor halmaz egy elemével pedig akkor van összeköttetés, ha az

oktató az adott szerepet legalább egy vizsgaalkalmon betöltheti. Az ábrán az „|S(Role)|:DEL ; |S(Role)|:(|E()|-|S(Role)|)*2p, |S(A)|:DEL” követelmény első, „|S(Role)|:DEL” tagja azt adja meg, hogy legalább annyi élnek kell kiindulna egy Student csúcsból a Student-Instructor kapcsolat felé, amennyi az adott Student csúcsból a Student-Role kapcsolat felé tart. Ez biztosítja, hogy legalább annyi oktatóval álljon kapcsolatban minden vizsgaalkalom, mint amennyi szerepkörre szükség van benne. A követelmény következő, „|S(Role)|:(|E()|-|S(Role)|)*2p” tagja azt jelenti, hogy egy Student csúcsból a Student-Role kapcsolat felé tartó élek száma fölött kettő büntetőpontot kell adni minden további kiinduló élért. Ezek a minimálisnál több oktatóért járó büntetőpontok. Végül az „|S(A)|:DEL” tag írja le, hogy egy Student csúcsnál legfeljebb annyi él lehet kiválasztva a szabály éltípusából, amennyi megegyezik a Student csúcsból a Student-A élcsúcsokba futó élek számával. Ez biztosítja, hogy ne legyen egy vizsgaalkalomra több oktató beosztva, mint amennyi szerep lehetséges az adott vizsgaalkalmon, hiszen akkor biztosan lenne olyan oktató, aki feleslegesen vesz részt. Az ábra második felén látható „1:DEL ; ” feltétel azt mondja ki, hogy ha egy adott Student-Instructor kapcsolatból egyetlen él sem indul a szerepekkel vett kapcsolatok felé, akkor törlésre kerüljön ez a Student-Instructor élpont, mivel egy olyan oktató nem vehet részt vizsgaalkalmon, aki nem tölt ott be semmilyen szerepet.

A következő feltétel azt mondja ki, hogy egy hallgató vizsgáján lehetőleg részt kell vennie a konzulensének, 5 büntetőpont számítandó minden olyan esetben, amikor ez nem teljesül.

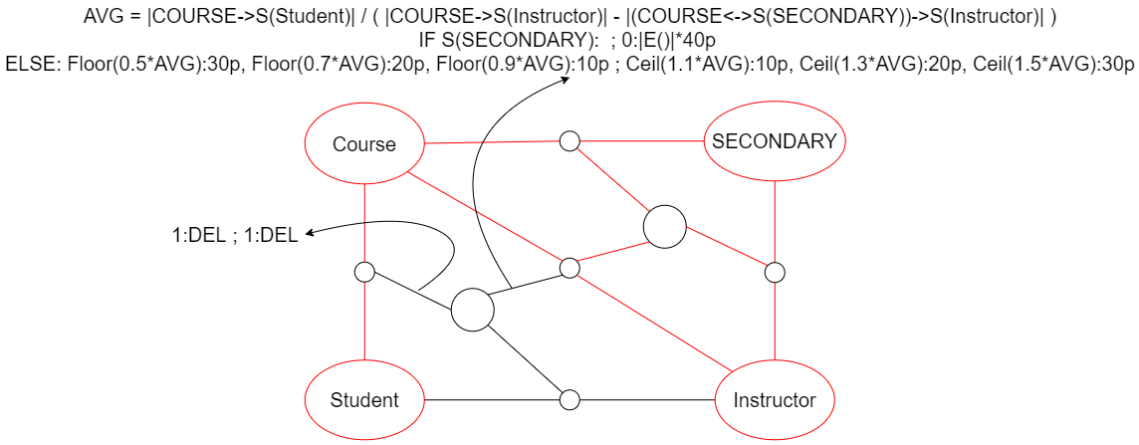


17. ábra - Konzulensre vonatkozó szabály

A 17. ábrán látható a konzulensre vonatkozó követelményt megadó háromszögkapcsolat. A Student-Instructor élpontból ki van választva az egyetlen hozzá tartozó Student-Instructor-CONSULTANT háromszögpont, mivel abban az esetben, ha egy oktató részt vesz a konzultáltja vizsgáján, akkor ott mindenképp betölti a konzulensi

szerepét is. Az „1:5p ; ” követelmény biztosítja, hogy 5 büntetőpontot jelentsen, ha valamely hallgatónál a konzulens szerepét egyetlen oktató sem tölti be, tehát nem vesz részt a konzulense a vizsgaalkalmán.

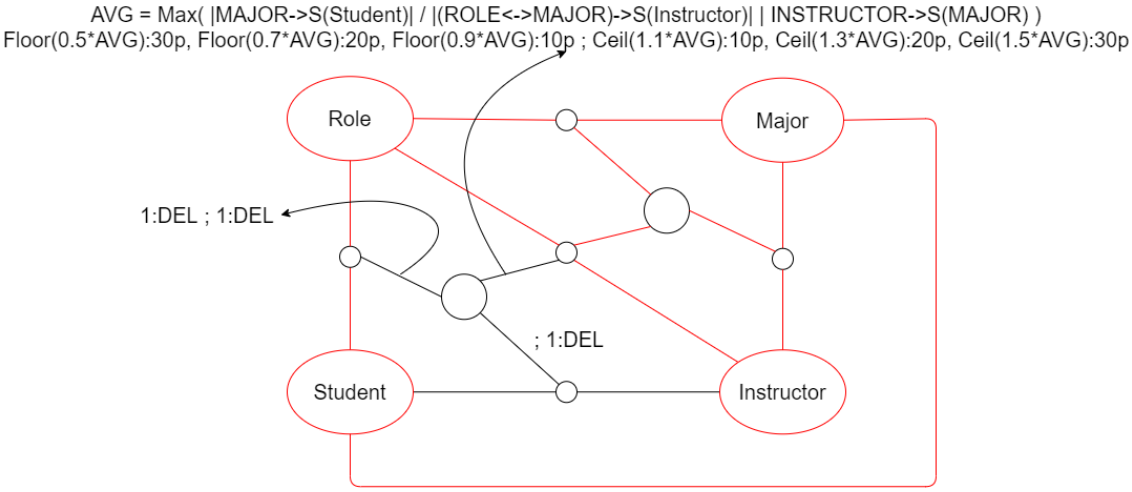
A vizsgatárgyakra vonatkozó első követelmény, hogy egy hallgatót egy tantárgyból pontosan egy oktatónak kell vizsgáztatnia. A bemeneti adatokban jelölve vannak olyan tanár-tantárgy párosok, amelyek nem ideálisak. Követelmény, hogy minden egyes vizsgaalkalomért 40 büntetőpont számítandó, amikor egy oktató nem ideális tantárgyból vizsgáztat. Végül pedig a vizsgáztató tanárok terhelésének kiegyenlítésére vannak megadva szabályok. Egy adott tantárgyból egy elsődlegesen vizsgáztató oktató optimális esetben annyi hallgatót vizsgáztat, amennyi a tantárgyból az egy elsődlegesen vizsgáztató oktatóra eső hallgatók száma. Ha az optimálistól 10%-nál jobban eltérő számú hallgatót vizsgáztat egy oktató, az 10 büntetőpontot jelent, 30%-nál több eltérés 20 büntetőpontot, és végül 50%-nál több eltérés 30 büntetőpontot eredményez.



18. ábra - Vizsgatárgyakra vonatkozó követelmények

A 18. ábra modellezi a vizsgatárgyakra vonatkozó követelményeket. Az AVG adja meg az optimális vizsgaszámot egy oktató-tantárgy párosra nézve. Az adott tantárgyból vizsgázó hallgatók számát kell elosztani a tantárgyból lehetségesen vizsgáztató oktatók számával, ahol a nem elsődlegesen vizsgáztató oktatókat nem kell beszámítani. Ha egy Instructor-Course kapcsolatból fut él a SECONDARY-vel közös kapcsolat felé, akkor a nem optimális oktató-vizsgatárgy párosra vonatkozó követelményeket kell alkalmazni, ellenkező esetben pedig az ideális terheléstől való eltérésért kell különböző büntetőpontszámokat adni.

Az elnököt, titkárt és belső tagot érintő követelmények a vizsgatárgyak követelményeihez hasonlóak. A különbségek a következők: Egy oktató egy hallgatónál ezen szerepkörök közül csak egyet tölthet be. Nincsenek nem optimális szerep-oktató párosítások, egy oktató vagy betölthet egy adott szerepet, vagy nem. A terhelések vizsgálata szakok között történik, egy szerepet egy oktató ideális esetben az egyes képzéseinek az optimális terhelései közül a maximális terhelés szerinti számú vizsgán tölti be.

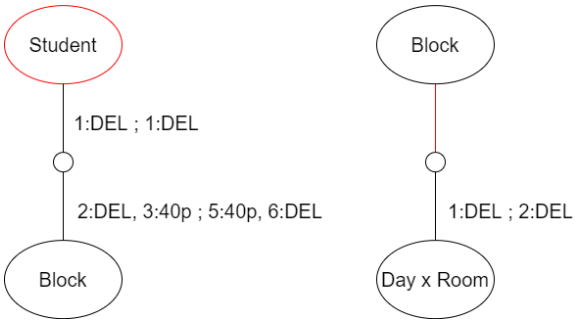


19. ábra - Oktatók vizsgaalkalmakon betöltött szerepköreire vonatkozó követelmények

A 19. ábra mutatja be az oktatók vizsgaalkalmakon betöltött szerepköreinek szabályait. Az optimális terhelés kiszámításához az adott Instructor csúcsból a különböző Instructor-Major kapcsolatok felé tartó éleinél kell az adott képzésbe tartozó hallgatók számát osztani a képzésben az adott szerepkört lehetségesen betöltő oktatók számával, és ezeknek a maximumát venni.

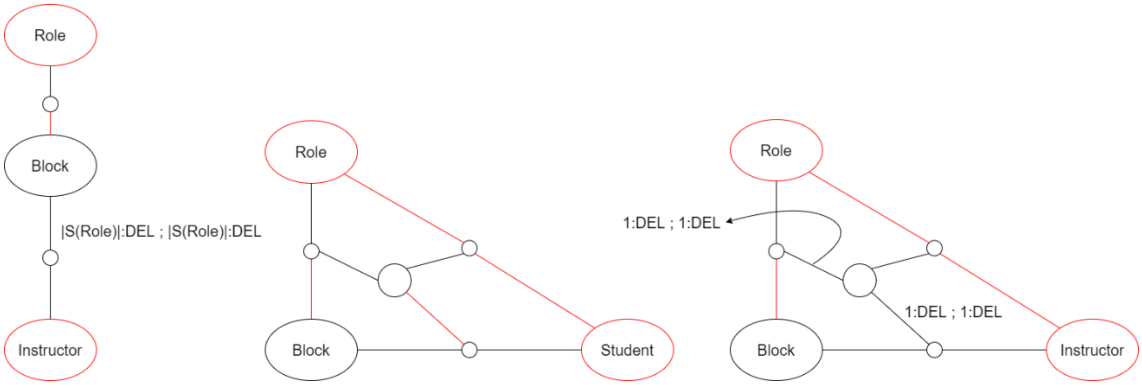
4.2.2 Blokkok és egész napos vizsgáztatási szekciók

Egy hallgatónak pontosan egy blokkban kell részt vennie, a vizsgák száma egy blokkon belül viszont változhat. Legalább 2 és legfeljebb 6 hallgató lehet egy blokkon belül, továbbá a 3-nál kevesebb vagy 5-nél több hallgató 40 büntetőpontot eredményez blokkonként. Egy teremben egy adott napon tartott vizsgáztatási szekció legalább 1 és legfeljebb 2 blokkot tartalmazhat.



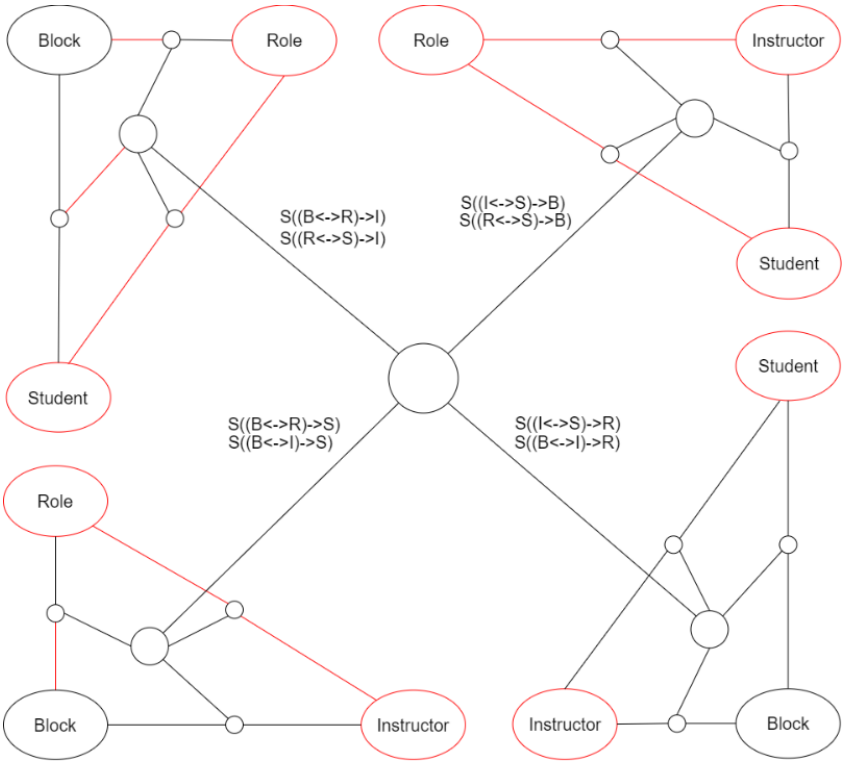
20. ábra - Az egy blokkon belüli hallgatók, és az egy teremben egy napon tartható blokkok számára vonatkozó követelmények

Blokkok vonatkozásában két oktatói szerepkörről beszélhetünk, a blokk elnökéről és titkáráról. Ezek a szerepek a vizsgaalkalmaknál is megjelentek, azonban az ottani belső tag szerepet nem kell blokkok szintjén figyelni, ugyanis arra nincsenek szabályok megadva blokkok vonatkozásában. Egy oktatót csak akkor kell egy blokkhoz társítani, ha ezen szerepek közül betölt pontosan egyet a blokkban. Egy blokkot csak akkor lehet megtartani, ha mindkét hozzá tartozó szerepet betölti egy-egy oktató, ezek a szerepek nem összevonhatók.



21. ábra – Blokkhoz tartozó szerepkörökre vonatkozó szabályok

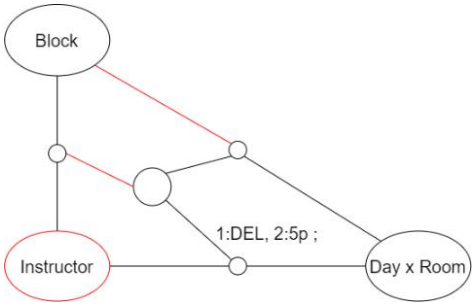
Fontos szabály, hogy egy oktató pontosan ugyanazt a szabályt töltse be a blokkban, mint a blokk minden vizsgáján.



22. ábra - Oktató blokkon belüli szerepének egységességét biztosító szabály

A 22. ábra szabálya az oktatók blokkon belüli szerepének egységességét biztosítja, amihez kiválasztást kényszerítő szabályokat alkalmaz. Ezek közül az „S(B<->R)->I)” jelentése nagy vonalakban a következő: Ha egy adott blokkon egy adott szerepkörre ki van választva egy oktató, akkor a blokk és szerepkör minden hallgatóra tekintett hármasából kiválasztandó a blokk-szerepkör-hallgató-oktató kapcsolat. A modellen található összes többi feltétel is ehhez hasonló jelentésű.

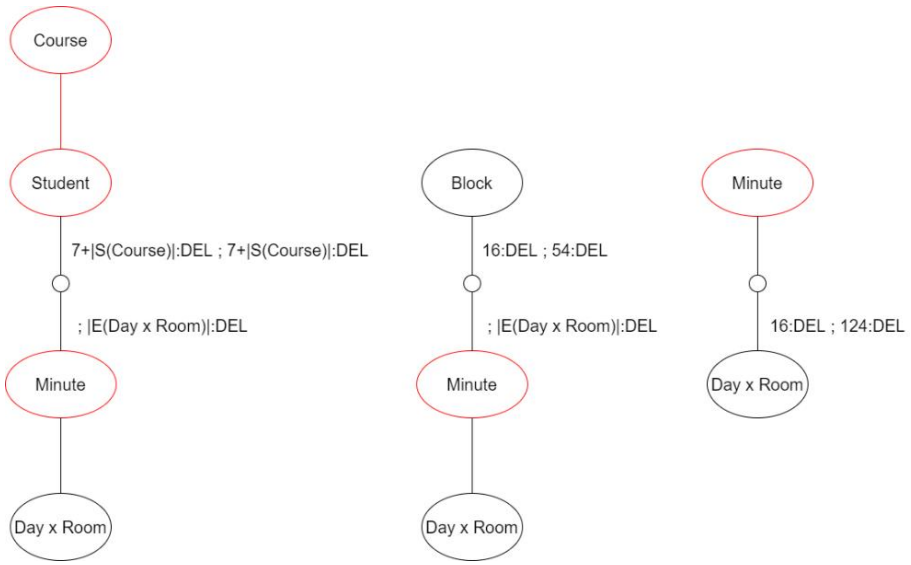
Ideális esetben egy teremben egy adott szerepet egész nap ugyanaz az oktató tölti be. Ha ez nem teljesül valamely szekcióban egy szerepre, az 10 büntetőpontot eredményez. Ez a szabály esetünkben ekvivalens azzal, hogy amennyiben egy teremben egy oktató csak egy blokkon vesz részt egy napon, úgy 5 büntetőpont számítandó.



23. ábra - Az oktatók nem egész napra történő beosztását büntető szabály

A 23. ábra kapcsán fontos megjegyezni, hogy az Instructor valamely csúcsa akkor van kapcsolatban a Day x Room egy csúcsával, ha az oktató részt vesz annak a szekciónak legalább egy blokkjában, de nem feltétlenül kell mindkét blokkjában részt vennie.

A különböző szekciók hosszát is szabályozzák feltételek. Ha egy hallgatónak 1 vizsgatárgya van, akkor pontosan 40 perces a vizsga hossza, ha pedig 2, akkor 45 perces. Ez alapján korábbi szabályok felhasználásával kiszámolható a minimális és maximális hossza a blokkoknak és az egy napon egy teremben tartott szekcióknak. Továbbá megadható még az is szabályként, hogy egy percben annyi blokk és vizsga tartható, amennyi terem rendelkezésre áll az adott percben.

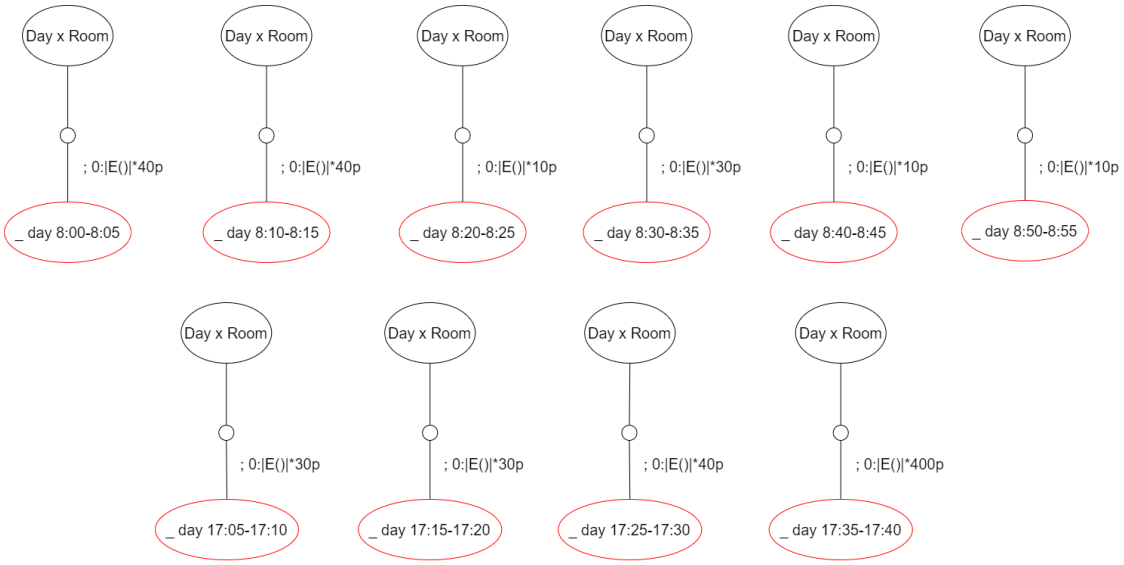


24. ábra - A különböző szekciók hosszára támasztott követelmények

A 24. ábrán megtekinthetőek a szekciók hosszait megadó szabályok. Mivel a Minute halmaz elemei 5 perc hosszúságú időintervallumokat reprezentáló csúcsok, ezért a modellben a Minute típussal vett kapcsolatok felé húzandó élek számát úgy kapjuk meg, hogy a korábban leírt követelményben megadott hosszokat leosztjuk öttel.

Követelmény, hogy vizsga nem kezdődhet korábban 08:00-nál, és nem végződhet később 17:40-nél. Továbbá különböző büntetőpontszámok vannak meghatározva a túl korai kezdésre vagy a túl kései befejezésre. A büntetőpontokat vizsgálhatjuk az egy napon egy teremben tartott szekciókra vonatkozóan. Mivel folytonosak a szekciók, ezért lehet azt vizsgálni, hogy mennyi olyan szekció van, amely bizonyos időpontnál korábbi vagy későbbi perccel áll összeköttetésben, és így egyszerűen meghatározhatóak a büntetőpontok. A pontos követelmények a 25. ábrán

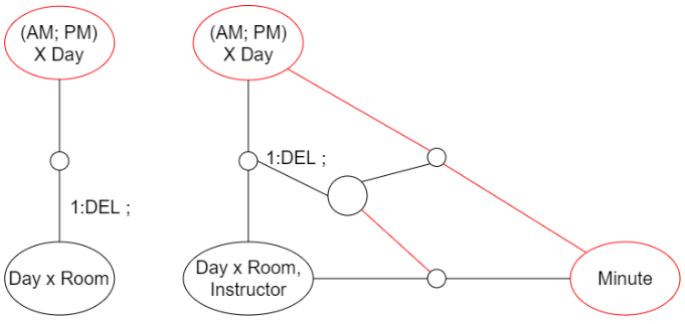
láthatók. A „_ day …” jelölés azt jelenti, hogy a záróvizsgáztatás összes napjáról kiválasztjuk a megadott 5 perces időtartományokat reprezentáló csúcsokat, és azokra vesszük fel az adott szabályt.



25. ábra - Vizsgák kezdetére és végére meghatározott büntetőpontok

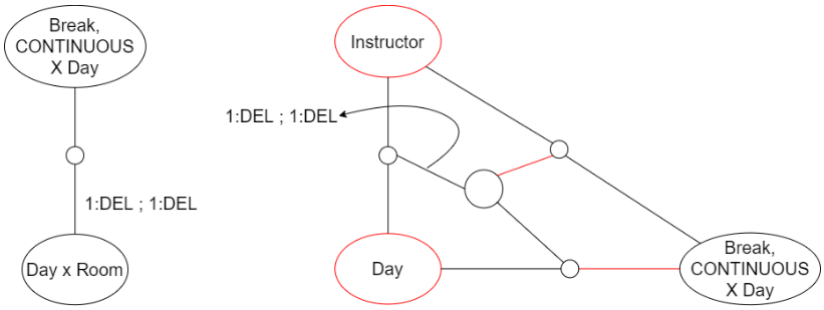
4.2.3 Szünetek

A szünetek követelményeinek megadásához szükség van napszakok használatára. Ez újabb szabályok kimondását követeli meg: Egy teremben a szekciót legalább egy napszakban tartják. Akkor tartanak szekciót egy napszakban, ha a napszak legalább egy percében tart a szekció. Továbbá oktató akkor vesz részt egy nap napszakában, ha az adott napszak legalább egy percében részt vesz.



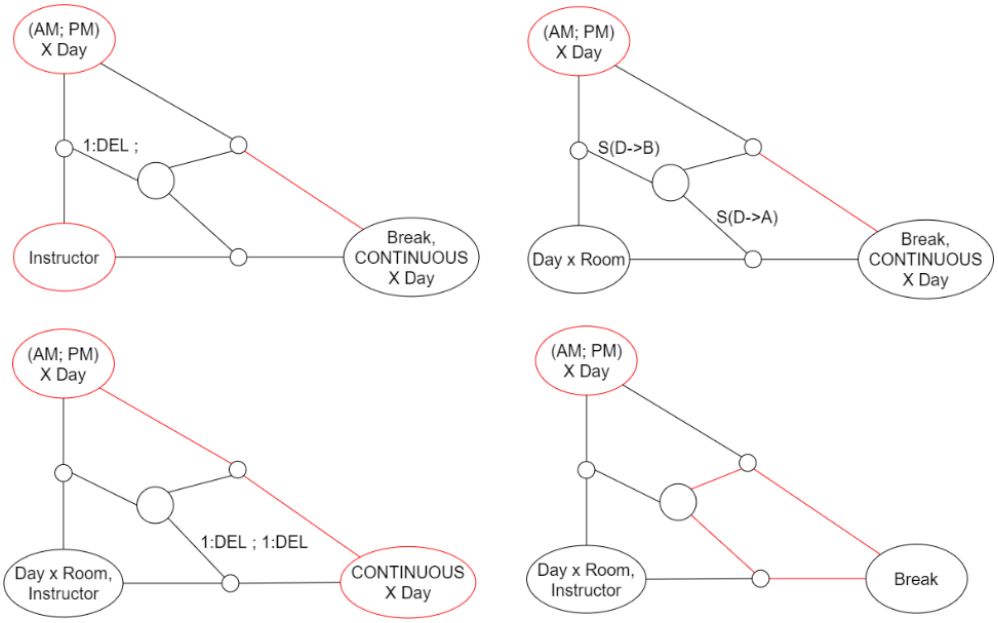
26. ábra - Napszakokra vonatkozó szabályok

Minden teremben tartott szekció vagy rendelkezik szünettel, vagy folytonos. Az oktatók napjaira is ugyanez a követelmény vonatkozik.



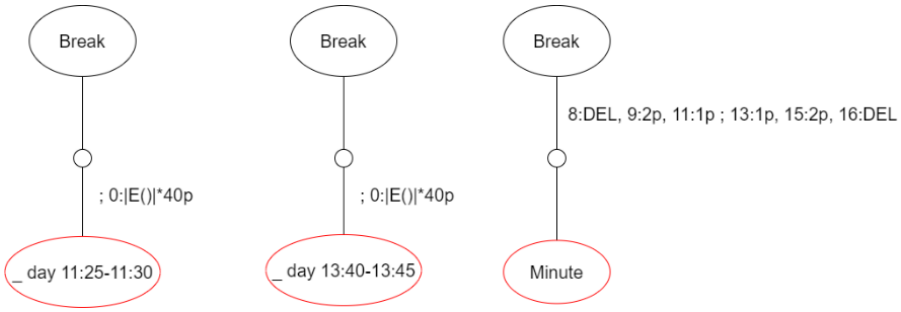
27. ábra - Szünetekre és folytonosságra vonatkozó alapvető követelmények

Az oktatók napszakjai, valamint a teremben tartott szekciók napszakjai biztosan részei a korábban említett entitások megfelelő napján vett folytonos beosztásának vagy szünetének. Egy oktató, vagy egy teremben tartott szekció egy napon pontosan akkor kell, hogy rendelkezzen szünettel, ha délelőttre és délutánra is be van osztva. Ezzel ellentétesen pedig akkor folytonos a beosztásuk egy adott napon, ha csak egy napszakban vesznek részt.



28. ábra - Szünet és napszak kapcsolatai

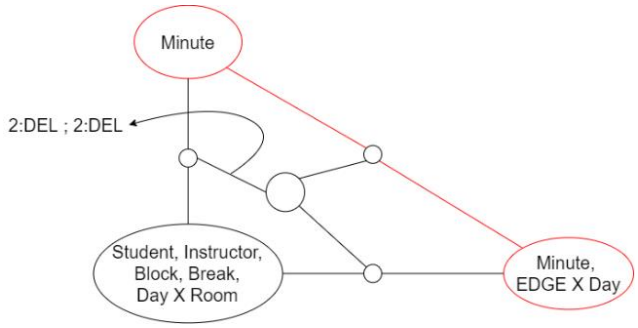
A szünetek túl korai kezdetére, túl kései végére, és nem optimális hosszára is vannak a különböző szekciókhoz hasonlóan büntetőpontok meghatározva, amelyek a 29. ábrán tekinthetők meg.



29. ábra - Szünet kezdetére, végére és hosszára támasztott követelmények

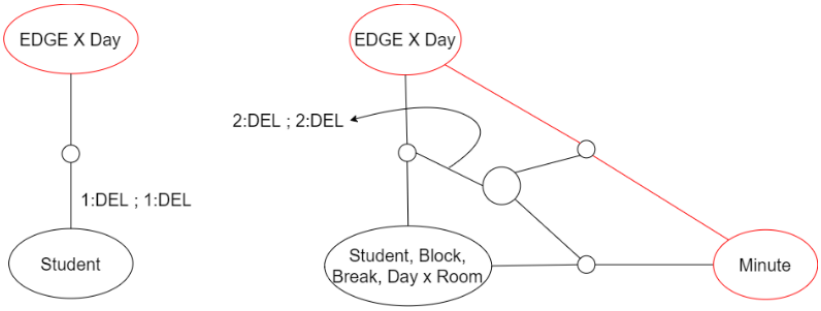
4.2.4 Különböző időtartományok

Egy időtartomány valamely perce vagy a tartomány két másik perce között helyezkedik el, vagy a tartomány szélén. Ez a szabály az időtartomány folytonosságának vizsgálatában játszik szerepet. A szabály használatához az szükséges, hogy a modellben minden perc össze legyen kötve minden vele szomszédos perccel.



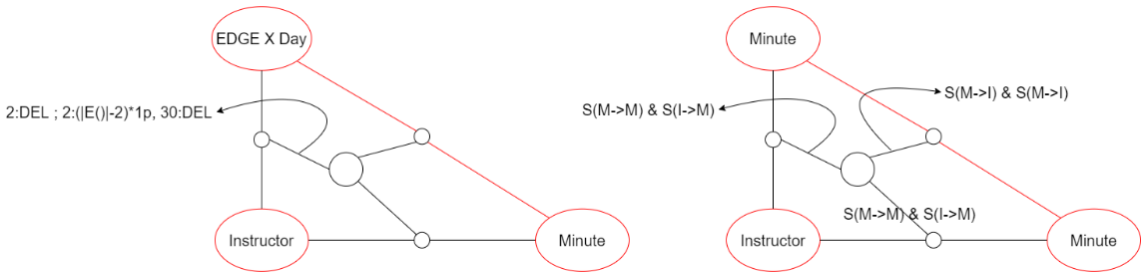
30. ábra - A különböző időtartományok széleit megadó szabály

Az egy napon egy teremben tartott szekciók, a szünetek, a blokkok és a vizsgaalkalmak időtartománya folytonos, így pontosan két szélük van. Minden hallgató vizsgaalkalmának kell lennie szélének valamely napon. A többi korábban említett időtartománynak is rendelkeznie kell éllel valamely napon, de erre nem kell felvenni külön szabályt, hiszen ezeknél az entitásoknál a bementi adatok alapján rendelkezésre áll, melyik napon tartják őket, így csak azon a napon lehet szélük.



31. ábra - Időtartományok folytonosságát biztosító feltételek

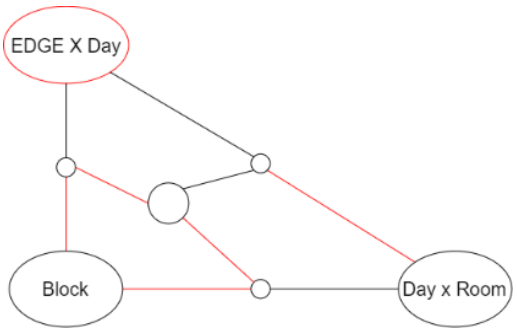
Az oktatók beosztásában lehetnek megszakítások egy napon belül is, viszont minden ilyen megszakítás 2 büntetőpontot eredményez, tehát az időtartományának minden kettőt meghaladó széléért egy büntetőpont számítandó. Meghatározható napi beosztásuk időtartománya széleinek maximális száma is. Még azt is ki kell kötni, hogy ha két egymással szomszédos percre beosztásra kerülnek, akkor ezen két perc között a beosztásuk folytonos legyen. Más időtartományoknál erre azért nem volt szükség, mert azok csak teljesen folytonosak lehetnek.



32. ábra - Oktatók beosztásának folytonosságára vonatkozó követelmények

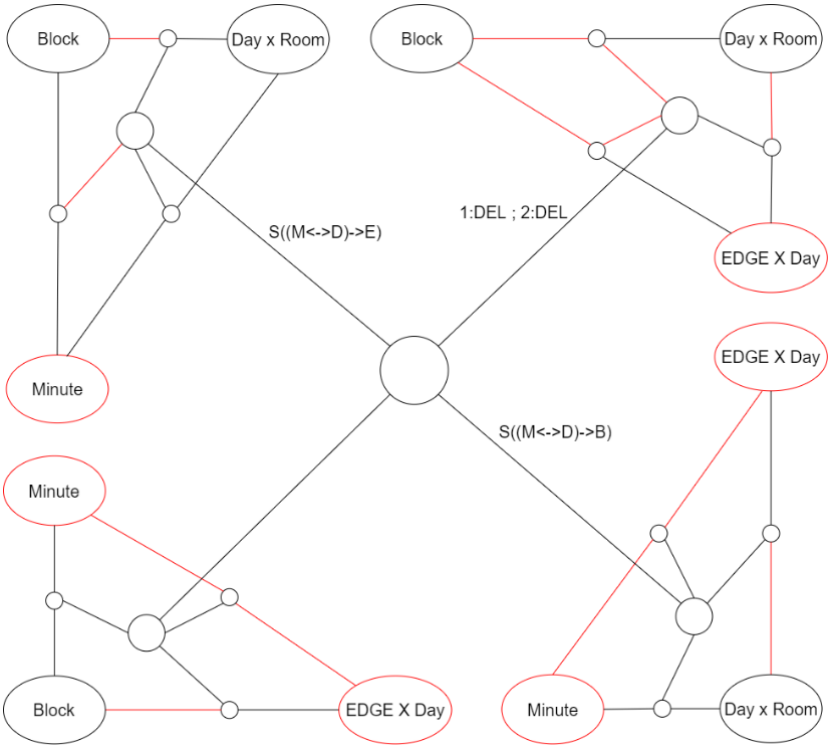
A 32. ábra második modellje biztosítja, hogy egy oktató folytonosan legyen beosztva két perc között, ha mindkét percre be van osztva. Tekintsük meg az egyik „S(M->M) & S(I->M)” feltételt. Ez azt mondja ki, hogy egy oktató-perc párosnál amennyiben a percből ki van választva egy másik perc, tehát időben egymást követik, továbbá az oktatóból is ki van választva az a perc, vagyis be van rá osztva, akkor az oktató-perc párosból kiválasztandó ez az oktató-perc-perc kapcsolat.

Egy teremben tartott szekció és a szekció egy blokkjának közös szélére vonatkozóan fel kell vennünk egy szabályt, így először ezen kapcsolatot is reprezentálni kell.



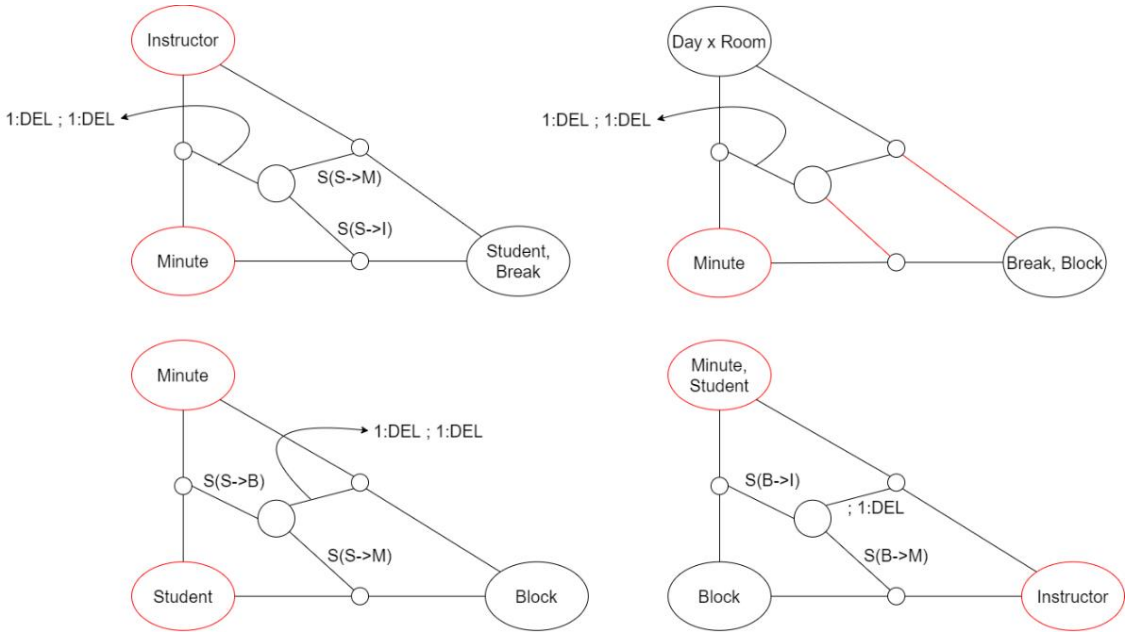
33. ábra - Egy napon egy teremben tartott szekció és annak egy blokkjának a közös szélét reprezentáló háromszögkapcsolat

Egy terem szekciójára és annak blokkjaira vonatkozó szabály a következő: minden blokknak és az azt tartalmazó szekciónak rendelkeznie kell legalább egy közös széllel. Ennek a szabálynak az a jelentősége, hogy ne lehessen két blokk közvetlenül egymás után, köztes szünet nélkül.



34. ábra – Blokkok és a blokkokat tartalmazó szekciók közös széleire vonatkozó szabály

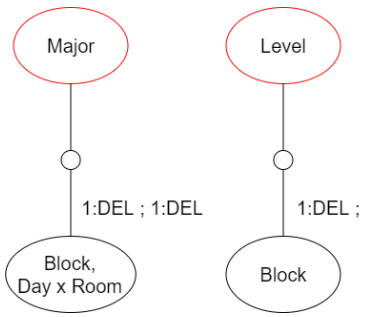
Egy oktató minden beosztott percében vagy egy vizsgaalkalmon van, vagy szüneten. Egy teremben tartott szekció minden percében vagy blokk van, vagy szünet. Egy blokk minden percében vizsgáztatás történik. Egy oktató egy percben vagy egy vizsgaalkalmon legfeljebb egy blokkba lehet beosztva. További fontos szabály, hogy bármely entitás csak akkor kerülhet beosztásra egy időtartományba, ha annak az időtartománynak minden részidőtartományára be van osztva. Ez a szabály egyedül az oktatók és az egy napon egy teremben tartott szekciók kapcsolatára nem vonatkozik, ugyanis ott megengedhető, hogy ne a teljes szekcióra legyen beosztva egy oktató.



35. ábra – Időtartományokra vonatkozó további szabályok

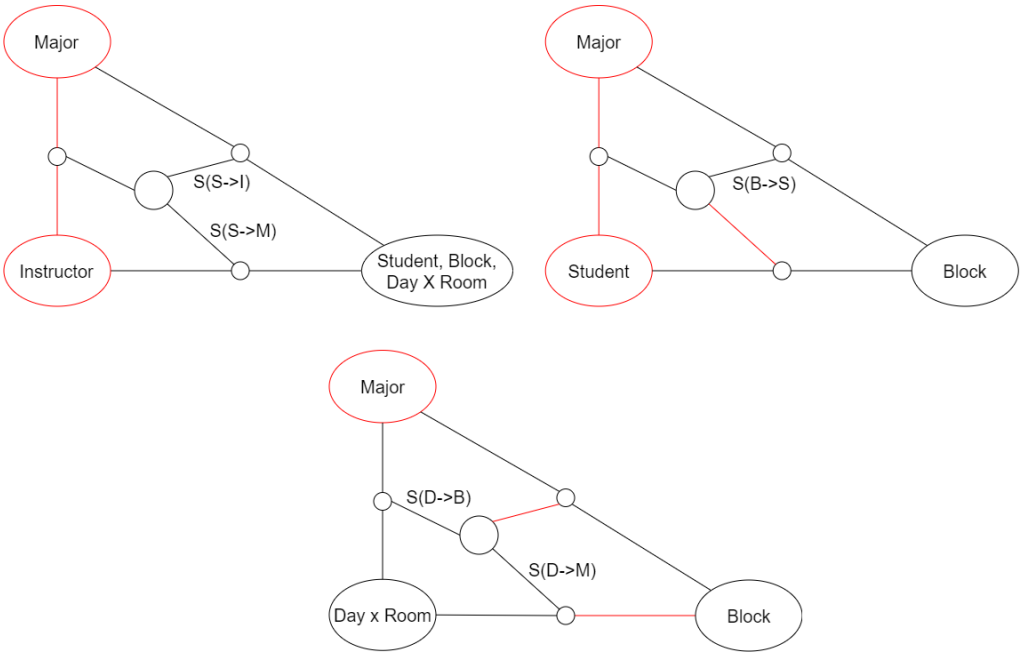
4.2.5 Képzések és képzési szintek

Az egész napos vizsgáztatási szekciók és blokkok pontosan egy képzésből tartandóak. A képzési szintekre vonatkozóan nincs ilyen követelmény, azonban egy blokkban legalább egy képzési szintből zajlanak a vizsgák, aminek későbbi feltételnél lesz szerepe. Az egész napos vizsgáztatási szekcióknál nincs képzési szintekre vonatkozó szabály, így ezeket a szekciókat képzési szintekkel nem kell társítani.



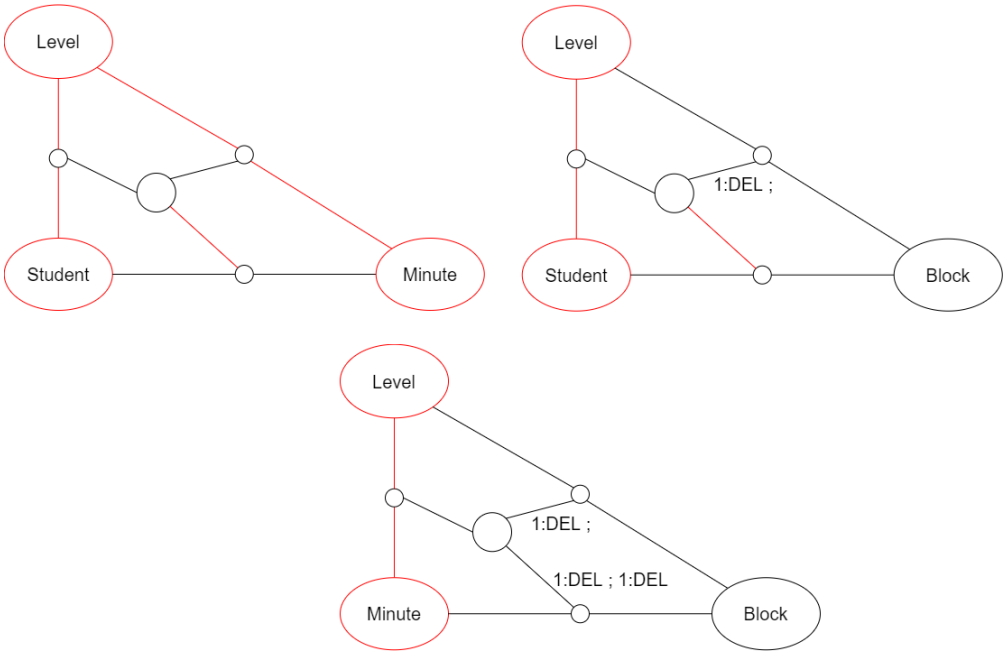
36. ábra - Képzésekre és képzési szintekre támasztott követelmények

Az oktatókra vonatkozóan meg van adva, hogy mely képzések vizsgáin vehetnek részt. Ha egy vizsgaalkalmon, blokkon vagy egész napos szekcióban részt vesznek, akkor annak a szekciónak a képzéséből tarthatniuk kell vizsgát. További követelmény, hogy egy vizsgáztatási szekciót abból a képzésből tartsák, amelyből a szekció részszekcióit.



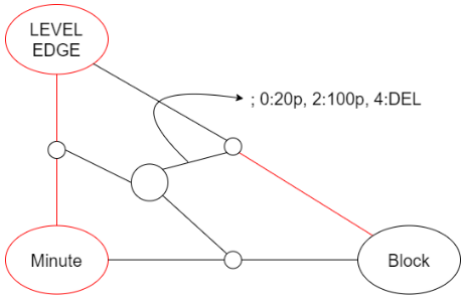
37. ábra - Képzések és különböző szekciók kapcsolatai

A képzési szinteket percek, vizsgaalkalmak és blokkok vonatkozásában vizsgáljuk, egy később ismertetett követelmény miatt. Egy percben bármely szint tartható, azok mindegyikével kapcsolatban áll. Egy blokkban azok a képzési szintek kerülnek megtartásra, amelyek a blokk legalább egy vizsgaalkalmában és legalább egy percében. A blokkok bármely percében pontosan egy képzési szintet lehet tartani.



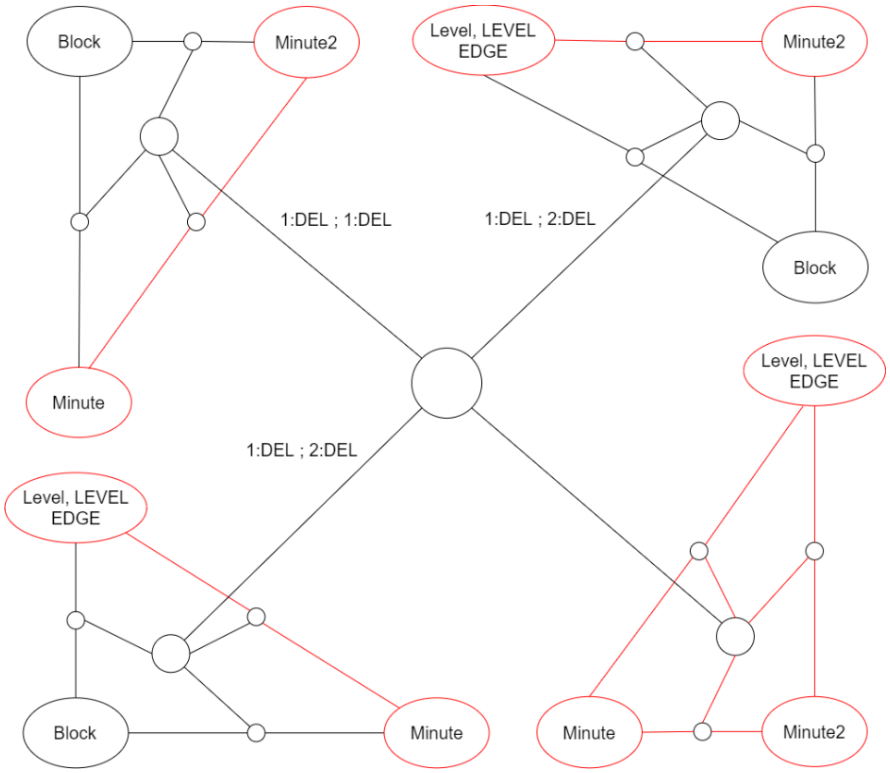
38. ábra - Képzési szintek és különböző szekciók kapcsolatai

Egy blokkon belüli képzési szintek váltásának számát minimalizálni kell. 1 váltás miatt 20 büntetőpontot kell számítani, 2 váltás 100 büntetőpontot eredményez és 3 váltás pedig szigorú követelményként tiltott. Ennek a követelménynek a vizsgálata ekvivalens azzal, hogy megvizsgáljuk, egy blokkon belül hány olyan perc van, amelyikben más képzési szint tartása történik, mint egy blokkon belüli vele szomszédos percben.

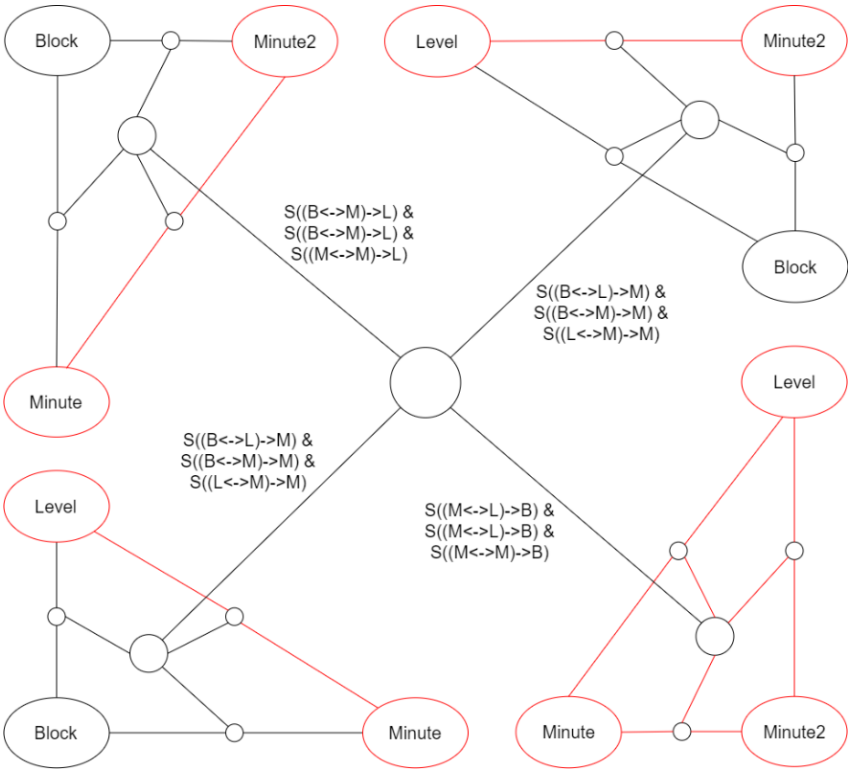


39. ábra - Blokkon belüli képzési szintek váltásainak számát büntető szabály

Egy blokk minden két szomszédos percéhez tartozik egy képzési szint, ha ezen percek képzési szintje megegyezik, vagy egy képzésiszint-váltás, ha eltér a képzési szintjük. Ezeket a szabályokat közösen írják le a 40. és 41. ábrán bemutatott modellek. A blokkok szélein lévő percek nem számítanak képzésiszint-váltásnak, csakis a blokkon belüli váltásokat vizsgáljuk.

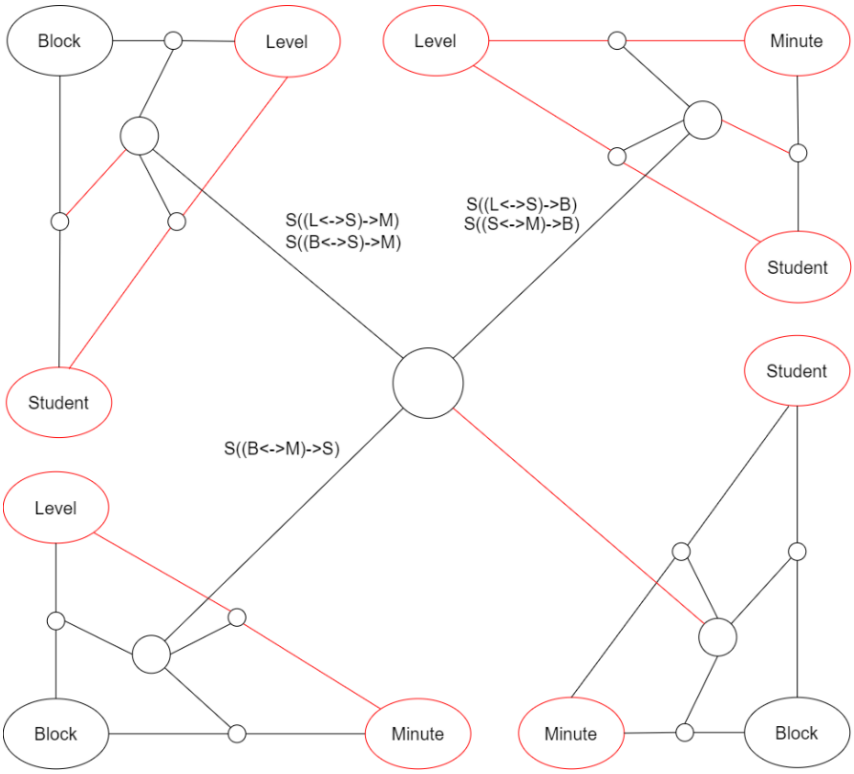


40. ábra – Blokkon belüli képzési szintek széleit megadó szabály



41. ábra – Folytonosságot biztosító szabály olyan blokkon belüli szomszédos perceknél, amelyeknél megegyezik a képzési szint

Végül pedig egy szabály biztosítja, hogy egy hallgató egy blokkon belüli percei ugyanazon képzési szinthez legyenek társítva, mint maga a hallgató.



42. ábra – Hallgatók, percek, blokkok és képzési szintek kapcsolata

5 Eredmények

A dolgozatban bemutatott állapotteret csökkentő saját módszerem teszteléséhez egy C++ implementációt is készítettem. A program az alapvető szabályokon kívül 1 mélységű próbálgatást is alkalmaz, következményszabályok nélkül.

A BME egy 100 fős heterogén hallgatói csoportjának záróvizsgabeosztás-készítésének problémáját használtam teszteléshez. A program egy Excel táblázatból olvasta be a különböző rendelkezésre álló adatokat. A táblázat első munkalapja az egyes hallgatókhoz tartalmazza a képzésüket, képzési szintjüket, konzulensüket és vizsgatárgyaikat. A második munkalapon érhetőek el az oktatók adatai: a betölthető szerepköreik, a képzések, amelyek vizsgáin részt vehetnek, továbbá a ráérésük órás bontásban az egyes vizsganapokra. Végül a harmadik munkalap az egyes vizsgatárgyakhoz adja meg, hogy mely oktatók vizsgáztathatnak belőlük, és mely oktatók vizsgáztathatnak ugyan, de nem ideális esetben.

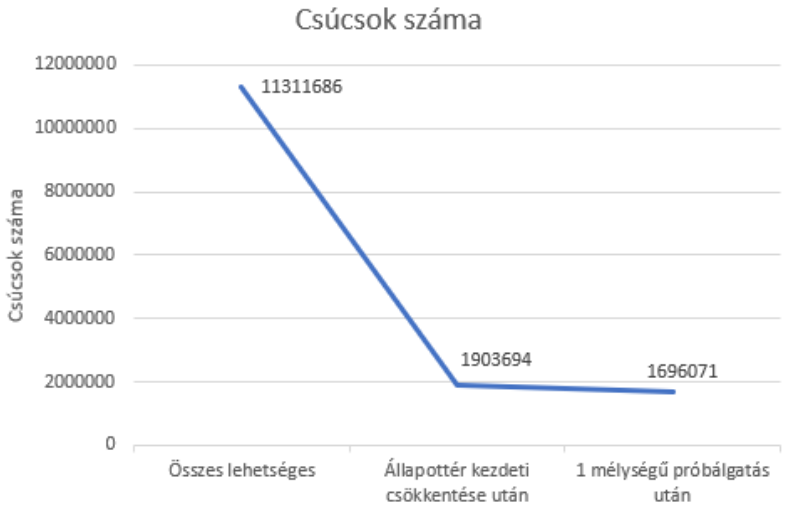
						2019.06.17. (hétfő)										2019.06.18. (kedd)									
Name	President	Member	Secretary	CS (I)	EE (V)	8:00	9:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	8:00	9:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00
Nagy Ákos					x																				
Ács Judit				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
Iváncsy Szabolcs		x		x																					
Ekler Péter		x		x																	x	x	x	x	x
Tóth Tibor				x		x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
Blázovics László		x		x	x	x	x	x	x	x	x	x	x	x	x										
Forstner Bertalan	x			x			x	x	x	x	x	x	x	x			x	x	x	x	x	x	x	x	
Kővári Bence András		x		x		x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

43. ábra - Oktatók adatainak egy részlete

Az 4. bekezdésben bemutatott módon formalizáltam a záróvizsgabeosztás-készítés problémáját. A program az összes szigorú és gyenge követelményt felhasználta. Az adatok beolvasása során is folyamatosan alkalmazta a szabályokat, a gyorsabb futás érdekében.

Egy 3,30 GHz-es i5-ös processzorral, valamint 16GB RAM memóriával rendelkező gépen futtattam a tesztet. Itt az adatok beolvasása és közben megtörténő állapottér-csökkentés 192 másodpercet vett igénybe. Az 1 mélységű próbálgatás 12 órán át tartott. A program futása során legfeljebb 1273 MB memóriát használt és átlagosan körülbelül 1100 MB-ot. Egy másodperc alatt 39,4 csúcs kipróbálását tudta elvégezni és 2786561 gráfon történő műveletet (kiválasztás/törlés).

Az állapottér méretére jó mérőszám a csúcsok száma, mivel a csúcsok a különböző lehetséges entitásokat és ezek lehetséges kapcsolatait reprezentálják. Egy adott problémánál a megadott típusokból, ezek kapcsolatából és a típusokban található entitások számából kiszámolható az összes lehetséges csúcs száma, ami a teljes állapottér mérete. A különböző állapotteret csökkentő lépések után maradt csúcsok száma összevethető ezzel, és így megvizsgálható, hogy mennyire csökkent az állapottér.



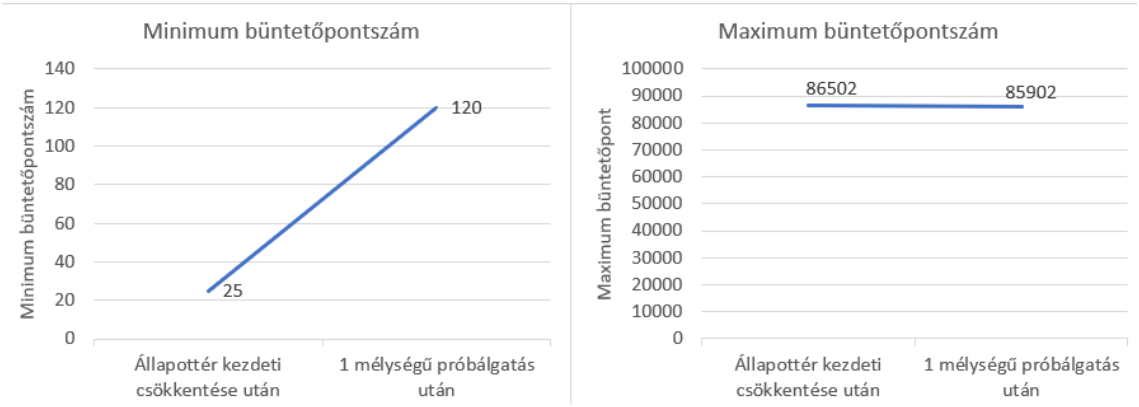
44. ábra - Csúcsok száma a program futásának egyes lépései után

A 44. ábra mutatja be a program futásának egyes lépései után maradt csúcsok számát. Látható, hogy az adatok beolvasásával párhuzamos állapottér-szűkítés jelentős mértékben csökkentette a pontok számát az összes lehetségeshez képest. Az 1 mélységű próbálgatások is tovább tudták csökkenteni az állapotteret, de ez már nem volt olyan nagy mértékű.

45. ábra - Csúcsok száma a futási idő függvényében

A 45. ábra a futási idő függvényében is ábrázolja a csúcsok csökkentésének mértékét. Látható, hogy az alapvető állapottér-csökkentéshez képest az 1 mélységű próbálgatás csak nagyon lassan hozott eredményt. Ezt később további következményszabályok, algoritmusok és optimalizáció segítségével lehet javítani.

A program minimum és maximum büntetőpontszámot is számított a gráfra, ezeket mutatja a 46. ábra. A próbálgatások jó eredménnyel végződtek a minimum büntetőpontszámra vonatkozóan, a maximumon viszont nem tudtak érdemileg változtatni. Azonban a minimum büntetőpontszám a legfontosabb, mivel ebből tudható meg, ha bizonyos gyenge követelményeket nem lehet kielégíteni a megadott adatokkal. Jelen esetben 5 olyan konzulens volt, aki a megadott ráérése következtében biztosan nem vehetett részt a konzultáltja vizsgáján. A maradék minimum büntetőpontszámot pedig az oktatók terhelésének eltérése okozta.



46. ábra - A gráf büntetőpontszámai az egyes futási lépések után

A program a futása során egy ellentmondást is felfedezett a bemeneti adatokban. Volt egy olyan oktató, aki más képzéshez tartozott, mint az egyik vizsgatárgya, így azon soha nem vehetett részt. Ezt javítottam, és jelen fejezetben már a helyes adatokon elért eredményeket ismertettem.

Mindezek után elmondható, hogy a tesztprogram képes volt az elvárt eredményeket hozni. Jelentősen lecsökkentette az állapotteret, és különböző ellentmondásokat is detektált a bemeneti adatokban és gyenge követelményekben.

6 Összefoglalás

Dolgozatomban egy saját gráf-alapú módszert mutattam be, amellyel beosztástervezési feladatok állapotterét lehet csökkenteni. A lecsökkentett állapottér más heurisztikák és megoldóalgoritmusok alapjául szolgálhat. A módszer által kizárt ellentmondásos állapotokkal egyáltalán nem kell foglalkozni, továbbá nem is készíthető olyan beosztás, amely ellentmond a felvett szigorú feltételeknek. Az egyes szabályos, de nem optimális beosztások azonosításához pedig büntetőpontszámokat lehet használni. Mindezek segítségével az is detektálható, ha a bemeneti adatokban vagy követelményekben ellentmondás van, és nem készíthető beosztás azok alapján.

A módszer működését részletesen ismertettem dolgozatomban. Különböző műveleteket és ezek szabályait dolgoztam ki, amelyek segítségével az adatok bevitele után megkaphatjuk a lecsökkentett állapotteret. Ezen felül további állapotteret szűkítő algoritmusokat is meghatároztam.

Modellt is készítettem, amellyel a különböző szigorú és gyenge követelmények megadhatók az állapotteret csökkentő módszer számára. A modell segítségével sikerült formalizálnom a BME egy 100 fős heterogén hallgatói csoportjának részére szóló záróvizsgabeosztás-készítés teljes problémáját.

Készítettem egy programot, amelyben a valós záróvizsgabeosztás-készítési feladatnál teszteltem a módszer működését. A program az állapotteret jelentős mértékben képes volt csökkenteni, továbbá ellentmondásokat is detektált az adatokban és követelményekben.

A jövőben a módszer kiegészíthető még több típusú követelmény kezeléséhez, hogy tetszőleges beosztástervezési feladatoknál használható legyen. Ezen felül több algoritmust is lehetséges definiálni hozzá, hogy még jobban tudja szűkíteni az állapotteret, vagy esetleg heurisztikák segítségével akár teljesen elkészítse a beosztást.

7 Irodalomjegyzék

[1] Rektori Kabinet Oktatási Igazgatóság, „A Szenátus X./10./2015-2016. (2016. VII. 11.) számú határozata A BME TANULMÁNYI ÉS VIZSGASZABÁLYZATÁRÓL," 1. Szeptember 2016.. [Online]. Available: https://kth.bme.hu/document/2061/original/BME_TVSZ_2016%20elfogadott_mod_20180801_web.pdf. [Hozzáférés dátuma: 21. Október 2021.].

[2] L. Sujbert, „BME VIK BSc szakdolgozat, záróvizsga, oklevél szabályzat a BME Tanulmányi és Vizsgaszabályzatába ágyazva," 7. június 2017.. [Online]. Available: https://www.vik.bme.hu/document/1343/original/BSc-ZV-170607.pdf. [Hozzáférés dátuma: 21. 10. 2021.].

[3] I. A. Chaudhry és A. A. Khan, „A research survey: review of flexible job shop scheduling techniques," *International Transactions in Operational Research,* pp. 551-591., 2016..

[4] N. Pillay, „A survey of school timetabling research," *Annals of Operations Research,* pp. 261-293., 2014..

[5] Vijindra és S. Shenai, „Survey on Scheduling Issues in Cloud Computing," *Procedia Engineering,* pp. 2881-2888., 2012..

[6] S. S. Panwalkar és W. Iskander, „A Survey of Scheduling Rules," *Operations Research,* pp. 45-61., 1977.

[7] J. K. Lenstra, A. H. G. Rinnooy Kan és P. Brucker, „Complexity of Machine Scheduling Problems," in *Annals of Discrete Mathematics 1,* Amsterdam, Netherlands, North-Holland Publishing Company, 1977., pp. 343-362..

[8] A. SCHAERF, „A Survey of Automated Timetabling," *Artificial Intelligence Review,* p. 87., 1999..

[9] S. Erdős, „Algorithm based on Hungarian Method," in *Proceedings of the Automation and Applied Computer Science Workshop 2019,* Budapest,

Magyarország, Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar, 2019., pp. 81-89..

[10] S. Erdős és B. Kővári, „Genetic Algorithm Based Solution for Final Exam Scheduling," in *MultiScience - XXXIII. microCAD International Multidisciplinary Scientific Conference,* Miskolc-Egyetemváros, Magyarország, Miskolci Egyetem, 2019., pp. 1-8..

[11] S. Erdős és B. Kővári, „Beosztástervezés lineáris programozással," *RENDVÉDELEM TUDOMÁNYOS FOLYÓIRAT,* pp. 47-62., 2020..

[12] S. Erdős és B. Kővári, „Algorithms Based on Analytic Learning Neural Networks for Final Exam Scheduling," in *SOR '21 Proceedings,* Ljubljana, Slovenia, BISTISK d.o.o., 2021., pp. 539-544..

[13] S. Erdős, „The problems of handling time inIP-based automatic scheduling," in *Proceedings of the Automation and Applied Computer Science Workshop 2020,* Budapest, Magyarország, Budapesti Műszaki és Gazdaságtudományi Egyetem Villamosmérnöki és Informatikai Kar, 2020., pp. 107-116..

[14] R. Varela és E. Soto, „Scheduling as Heuristic Search with State Space," in *Advances in Artificial Intelligence — IBERAMIA 2002,* Berlin, Germany, Springer, 2002., pp. 815-824..

[15] R. Cheng, M. Gen és Y. Tsujimura, „A tutorial survey of job-shop scheduling problems using genetic algorithms—I. representation," *Computers & Industrial Engineering,* pp. 983-997., 1996..

[16] J. Wang, X. Fan, C. Zhang és S. Wan, „A Graph-based Ant Colony Optimization Approach for Integrated Process Planning and Scheduling," *Chinese Journal of Chemical Engineering,* pp. 748-753., 2014..

[17] „Javító út keresés páros gráfokban, magyar módszer," [Online]. Available: http://www.math.u-szeged.hu/~hajnal/courses/MSc_Grafelmelet/grafelmelet/magyar.htm. [Hozzáférés dátuma: 26. október 2021.].

[18] S. Erdős, „Automatizált záróvizsga beosztás készítése," in *Tudományos Diákköri*

Konferencia, 2018..

[19] S. Erdős és B. Kővári, „Algorithms for final exam scheduling,” 18. július 2018.. [Online]. Available: https://github.com/BenceKovari/Schedule/blob/master/Fitness.md. [Hozzáférés dátuma: 21. 10. 2021.].