CSOPORTKIVÁLASZTÁSI MÓDSZEREK VIZSGÁLATA RUGALMAS PROJEKTTERVEZÉSI MEGKÖZELÍTÉSEK ESETÉN[1]

KOSZTYÁN ZSOLT TIBOR[a], HARTA PÉTER[a,b], SZALKAI ISTVÁN[c], OBERMAYER NÓRA[d]

[a,c,d]*Pannon Egyetem,* [a]*Kvantitatív Módszerek Intézeti Tanszék,*
[c]*Matematika Tanszék,* [d]*Szervezési és Vezetési Intézeti Tanszék, Veszprém*
[b]*Continental Teves Automotive Hungary Kft., Veszprém*

Napjainkban a legtöbb szoftverprojekt rugalmas, például agilis, extrém vagy hibrid környezetben valósul meg. Ezekben a környezetekben a projektekben szereplő tevékenységek végrehajtásához prioritásokat rendelnek. A kötött végrehajtási struktúra helyett pedig általában a rugalmas végrehajtás a jellemző, ahol a tevékenységek végrehajtási sorrendje a projekt előrehaladtával változhat. Ráadásul a rugalmas megközelítéseket egyre inkább alkalmazzák nem szoftverprojektek esetén is. A legtöbb szoftverprojekt-ütemezési probléma (angolul: Software Project Scheduling Problem, SPSP), amely egy tágabb problémakör, a projektütemezési probléma (Project Scheduling Problem, PSP) része, fix projektvégrehajtási struktúrát feltételez. Ha ezeket a hagyományos módszereket használjuk, nem tudjuk modellezni és így megérteni, hogy bizonyos rugalmas, pl. agilis irányelvek miért elengedhetetlenek, és hogyan segítik a projektmenedzsmentet. Kutatásunk bemutatja, hogy egy mátrixalapú projekttervezési megközelítés hogyan modellezheti a csoportszerepeket és a munkavállalók közötti szinergiákat. A szinergiaalapú szoftverprojekt-ütemezés problémáját kibővítve a tanulmány bemutatja az autonóm csoportszerepek kiválasztásának hatásait a DISC viselkedéstípusokon keresztül. A javasolt modell képes kezelni (1) a rugalmas projekteket, (2) a munkavállalók közötti szinergiákat és (3) a kemény és a puha készségek (soft and hard skill) közötti különbséget. Tanulmányunk hidat képez az operációkutatási modellek, a csoportszerepek modellezése, valamint a rugalmas projektütemezési megközelítések között. A javasolt módszert egy valós vállalati esettanulmányon keresztül értékeljük. Az eredményeket összehasonlítjuk a meglévő, nem szinergiaalapú projektütemezési módszerekkel. Vizsgálataink eredménye nemcsak azt mutatja meg, hogy a munkavállalók közötti pozitív szinergia csökkentheti a projekt átfutási idejét, hanem azt is, hogy a legjobb megoldást mely csoportszerep kiválasztása adja.

Kulcsszavak: Rugalmas projektütemezési megközelítések, szinergiaalapú modellezés, csoportszerepek, DISC viselkedési stílusok, csoportszerep-kiválasztás. *JEL-kód:* C88, D83, M15, O32

[1]Beérkezett 2024. június 20. DOI: https://doi.org/10.15170/SZIGMA.55.1219. E-mail: kzst@gtk.uni-pannon.hu.

1 Bevezetés

Amíg manapság még az Ipar 4.0-ban az információtechnológia (Information Technology (IT)) szerepe, és így a szoftverprojektek szerepe is növekszik, addig a jövőben az Ipar 5.0 hozhatja el azt az időszakot, amikor a szoftvereket, vagy legalábbis azok egyes elemeit már nem emberek, hanem gépek készítik. Ma még azonban a szoftvereket szorosan együttműködő fejlesztők készítik [42, 54]. A szoftverfejlesztés fontosságát tovább hangsúlyozza a széles körben vizsgált [51], ún. szoftverprojekt-ütemezési probléma (Software Project Scheduling Problem, SPSP), amely egyidejűleg foglalkozik az emberi erőforrások helyes kiválasztásával, valamint az emberierőforrás-igényes szoftverprojektek ütemezésével. Maga ez az operációkutatási probléma nagyon hasonló az ún. *különféle képességet igénylő erőforrás-korlátos projekt-ütemezési problémához* (Multi-Skill Resource-Constrained Project Scheduling Problem (MS-RCPSP)), amelynek hasonló korlátai és hiányosságai vannak [4, 56]. Mind az SPSP, mind az MS-RCPSP esetében a feladat időtartama függ a munkavállalók készségétől [2, 13, 38], valamint megadhatnak a feladatok között fix [16, 31] vagy előre meghatározott [9] prioritásokat. Azonban egyik probléma sem veszi figyelembe a munkavállalók közötti együttműködést vagy más néven a szinergikus kapcsolatokat. Ezek a megszorítások erősen korlátozzák e módszerek gyakorlati alkalmazhatóságát, különösen olyan rugalmas, autonóm projektkörnyezetben, ahol a szoftverfejlesztő mérnököknek együtt kell dolgozniuk.

A mai állandóan változó világban már az előre rögzített projekttervek kevésbé tarthatók szoftverfejlesztési projektek esetében [57]. Ütemezési szempontból az *1. táblázat* foglalja össze a hagyományos és rugalmas projekttervek eltérő ismérveit.

Projekttervezési módszerek	Főbb jellemzők			
	Projektstruktúra	Új tevékenység	Végrehajtási módok	Korlátok
Hagyományos	Kötött	Nem megengedett	Megengedett	Kötött
Agilis	Rugalmas	Nem megengedett	Nem kezelt	Kötött
Extrém	Rugalmas	Megengedett	Nem kezelt	Rugalmas
Hibrid	Rugalmas	Megengedett	Megengedett	Opcionális

1. táblázat. Hagyományos és rugalmas tervezési módszerek összehasonlítása

Wysocki [57] 2011-es kutatásában azt találta, hogy a hibrid megközelítést, mely magában foglalhatja akár a hagyományos, akár a rugalmas (agilis, extrém) projekteket, leszámítva a szoftverprojektek hagyományos, kötött projektstruktúrát feltételező megvalósítása mintegy 20%-a az eseteknek. 70%-a agilis és 10%-a extrém. Wysocki [57] későbbi megismételt kutatása már azt mutatta, hogy tisztán rugalmas, vagy tisztán tradiconális projektmegközelítés ritkán fordul elő, ugyanakkor az ilyen hibrid megközelítésekben is azonosítható az agilis, vagy a tradicionális megközelítések dominanciája.

Az 1. táblázat összehasonlítja a hagyományos és a rugalmas ütemezési eljárásokat. Nem állítjuk, hogy a hagyományos projektmenedzsment területén ne szembesültek volna azzal, hogy a projekten akár menet közben ne kellene

változtatni, ami a projekt struktúrájára is hatással lehet. Ugyanakkor a módszerek ismertetése során kötöttnek tételezték fel a tevékenységek végrehajtási struktúráját. Emiatt a hagyományos módszerek ezekben a megközelítésekben nem voltak alkalmazhatók. A hagyományos módszerek nem támogatják az új, nem várt tevékenységek ütemezését sem, ekkor ugyanis az ütemtervet szintén módosítani kell. A rugalmas megközelítések ugyanakkor eleve rugalmasan kezelik a projekttervet, feltételezve azt, hogy a tevékenységek többféle sorrendben is végrehajthatók. A rugalmas megközelítések közül az agilis módszerek nem kezelik a többféle végrehajtási módot. E megközelítés az új, nem várt tevékenységeket csak a következő projektszakaszban (sprintben) veszi figyelembe, míg az extrém megközelítések megengedik az előre nem tervezett tevékenységek végrehajtását is. Ugyanakkor csak a hibrid megközelítést támogató módszertan az, amely képes mind a tevékenységek többféle végrehajtási módját, mind a projektstruktúra rugalmasságát, mind pedig az új, nem várt tevékenységeket figyelembe venni, és ezáltal meghatározni az optimális ütemtervet.

Az agilis és hibrid megközelítéseket segítő projekttervezési módszereknek már mindenképpen figyelembe kell venniük a végrehajtási struktúrák rugalmasságát, úgymint a tevékenységek prioritását, valamint a rákövetkezési relációk feloldhatóságát [5, 22]. A megrendelő, együttműködve a fejlesztőkkel prioritásokat fogalmaz meg a feladatok végrehajtásával kapcsolatosan [43], amely prioritási lista alapján eldönthető, hogy egy feladatot végül az adott vagy egy későbbi fejlesztés során hajtanak-e végre [1]. Emellett a rugalmas, ezen belül is kiemelten az agilis megközelítések nagymértékben támaszkodnak a fejlesztők közötti folyamatos kommunikációra és együttműködésre, amelynek szinergikus hatása jelentősen növeli a szoftverfejlesztési folyamat eredményességét [55]. Az általunk javasolt mátrixalapú módszer mindegyik megközelítést képes kezelni, ugyanakkor az esettanulmányunk agilis projektkörnyezetből származik, így a modellünket is ennek megfelelően alakítottuk ki. Ilyen környezetben a végrehajtási sorrendje változhat. A tevékenységek prioritásuk függvényében kerülnek be egy-egy részprojektbe (ún. futamba, vagy sprintbe), ugyanakkor nem engedjük meg a több lehetséges végrehajtási módot, valamint ha új probléma, vagy megoldandó feladat merül fel, akkor azt csak a soron következő sprintbe ütemezhetjük be.

Az SPSP fő gondolata az volt, hogy az emberi erőforrásokhoz olyan készségeket rendeljenek, amelyek lehetővé teszik számukra egy feladat elvégzését [2]. Később az SPSP kiterjesztésével [51] a kutatók már azt vizsgálták, hogy ezen készségek szintje hogyan befolyásolja a feladat elvégzéséhez szükséges időt. Vannak olyan modellek, amelyeket elsősorban hosszabb projektek esetén érdemes alkalmazni, ahol a készségek idővel változhatnak [49, 51]. A legtöbb modellben a készségek szintje közvetlenül befolyásolja a feladat időtartamát. A készségek magasabb szintje rövidebb időtartamot eredményez, ugyanakkor ezek a készségek nem additívak. A nem összegezhető készségek helyett Kosztyán és mtsai. [23, 26] a *készség-kapacitás* használatát javasolták. Például két munkavállaló programozási vagy tesztelési készségei nem adhatók össze, továbbá nem adhatók össze a nyelvi készségek sem, ugyanakkor a kódolt és

ellenőrzött programsorok száma, a kiértékelt funkciók száma, vagy éppen az elkészített angol nyelvű dokumentációk száma összeadható. Ezenkívül sokkal könnyebb a készség-kapacitásra vonatkozó minimális megkötéseket meghatározni, mint magukra a készségekre. Ez a modell erősen támogatja az agilis megközelítéseket, ahol az egyik követelmény az agilis kiáltvány [6] alapján a keresztfunkcionalitás, azaz, hogy az erőforrások hasonló készségekkel rendelkezzenek egymás helyettesítésére. Nem szabad elfelejteni, hogy általában a megrendelőt csak a végtermék érdekli, sokkal kevésbé az, hogy végül ki végzi el a feladatokat, és a munkavállalók hogyan osztoznak a feladatok teljesítésében. Mindazonáltal meg kell jegyezni, hogy az ilyen összegezhető kapacitások elsősorban csak a kemény készségek (hard skills) esetében modellezhetők. A puha készségek, például a kommunikációs készségek esetében nem, vagy csak nagyon nehezen lehet készség-kapacitást meghatározni. Véleményünk szerint így mindkét elképzelésnek helye van a modellezésben, ezért a javasolt modellben a készségek bináris és ordinális szintjei, valamint a kardinális készség-kapacitás is modellezhető.

Ha az operációkutatás területéről átlépünk a menedzsment területére, akkor azt tapasztalhatjuk, hogy a munkavállalók közötti szinergiát és ennek a szinergiának a munkavállalók teljesítményével való kapcsolatát már széles körben tanulmányozták [lásd pl.: 17, 27, 28], ugyanakkor eddig Kosztyán és mtsai. [23] által készített tanulmány az egyetlen, amely a hagyományos szoftverprojekt-ütemezési feladatot kiterjesztve figyelembe veszi a munkavállalók közötti szinergiát is, szakít a fix végrehajtási strultúrák és a nem összeadható készségek megkötéseivel. A tanulmány az SPSP egy új formáját javasolta, amelyet Synergy-based Software Project Scheduling Problem (SSPSP)-nek neveznek. Kosztyán és mtsai. [23] mesterségesen előállított projekteket és szinergiahálózatokat vizsgáltak a szinergiák projektütemezésre gyakorolt hatásainak meghatározására. Ezt a megközelítést azonban mind ez idáig nem vizsgálták valós, gyakorlati példán keresztül. A gyakorlati megvalósítás egyik legnagyobb kihívása ugyanis a munkavállalók közötti szinergiahálózat meghatározása és azok teljesítményre gyakorolt hatásának számszerűsítése. A gyakorlat azt mutatja, hogy sokszor a készség-kapacitások meghatározása különösen puha készségeknél meglehetősen nehézkes, ezért egy olyan hibrid modellt javaslunk, amely képes a munkavállalók viselkedési stílusán keresztül nemcsak a szinergiák megbecslésére, de az egyes munkavállalók puha készségeinek számszerűsítésére is.

Bár a szoftverprojektek ütemezésével foglalkozó szakirodalom még csak kis részben vette át a menedzsment területén már régóta figyelembe vett munkavállalók közötti szinergikus hatásokat, a hagyományos SPSP kiegészült a különböző személyiségtípusok figyelembevételével [50]. Számos tanulmány [29, 44, 45] kimutatta, hogy a szinergiát erősen befolyásolja a személyiségtípus és a csoportszerep. Belbin [7] szerint a csoportszerepek és a szinergiák között kapcsolat figyelhető meg. Közelebb kerülve a szoftverprojektek menedzseléséhez további tanulmányok [15, 48, 52] vizsgálták a személyiségtípusok és a páros programozás sikeressége közötti kapcsolatot [40]. Belbin azt javasolta, hogy a csoportokban lévő különböző feladatokat ellátó csoporttagokat vá-

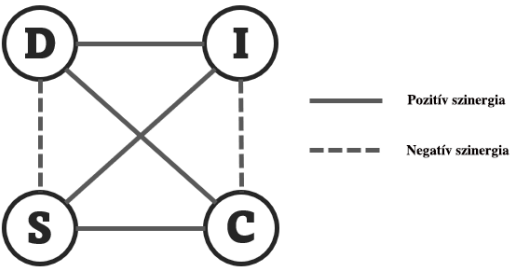
lasszuk szét úgynevezett Belbin-féle csoportszerepekre, és rendezzük őket hármas csoportokba vagy triádokba [7, 8]. Megállapítható, hogy a különböző Belbin-féle csoportszerepekből álló csoport kialakítása csökkentheti a konfliktusok kialakulásának lehetőségét, és növelheti a csoport tagjai közötti munkakapcsolatok erősségét [35, 7, 8]. Hardiman [20] és Capretz [11] a szoftverfejlesztő csoportok tagjainak leggyakoribb személyiségtípusait tanulmányozta, Capretz és Ahmed [12] pedig a széles körben ismert Myers-Briggs személyiségmodell [37] segítségével vizsgálta a személyiségtípusok hatását a szoftverfejlesztő csoportok teljesítményére.

Mielőtt az agilitás módszertanát kidolgozták volna a szoftverfejlesztési projektekre, illetve mielőtt felismerték volna a kis létszámú csoportok fontosságát, leginkább a 16 különböző személyiségtípust megkülönböztető Myers-Briggs modellt alkalmazták a Belbin-féle módszer helyett, mivel az alkalmasabbnak bizonyult a nagyobb projektcsoportok vizsgálatára. Több tanulmány [18, 35, 36] is foglalkozik a Belbin-féle, valamint a Myers-Briggs-féle csoportkiválasztás [11, 44] módszerével, ugyanakkor kisebb csoportok kiválasztása esetén ezek a módszerek kevésbé alkalmazhatók. Kisebb figyelmet kap viszont a viselkedési stílusok vizsgálata, mint pl. a DISC viselkedésistílus-rendszer, amely rugalmasabb a viselkedési stílusok kis száma (mindösszesen 4) miatt. Egyszerűsége és népszerűsége okán a DISC viselkedési stílusokkal kapcsolatos tanulmányok száma az elmúlt években fokozatosan nőtt, és a kis agilis csoportok kiválasztására is inkább ezt a módszert alkalmazzák [32, 39].

A DISC viselkedési stílus modell, amelyet Marston [34] 1928-ban dolgozott ki, az egyéneket viselkedési stílusuk alapján különbözteti meg. A DISC betűszó jelöli az egyes stílusokat, amely az extrovertált – introvertált és a feladatorientált – kapcsolatorientált dimenzióin keresztül csoportosítja az embereket. A modell szerint egy ember viselkedése lehet domináns (D), befolyásoló (I), kitartó (S) vagy szabálytisztelő (C). Minden viselkedési stílusnak egy-egy szín feleltethető meg: piros a domináns (D), sárga a befolyásoló (I), zöld a kitartó (S) és kék a szabálytisztelő (C). Az egyének túlnyomó többsége több viselkedési stílussal is leírható, ugyanakkor mindenkinek van egy ún. elsődleges viselkedési stílusa. A *2. táblázat* foglalja össze az elsődleges viselkedési stílusok jellemzőit.

Viselkedési stílus	Jellemzés	Főbb tulajdonságok
Domináns	extrovertált, feladatorientált, a csoport vezetője, felelősséget vállal	aktív, erőteljes, konfrontálódó, makacs
Befolyásoló	extrovertált, kapcsolatorientált, a csoport lelke, fenntartja a motivációt	barátságos, optimista, könnyed, labilis
Kitartó	introvertált, kapcsolatorientált, a csoport lelke, fenntartja a stabilitást	csapatjátékos, megbízható, lojális, visszahúzódó
Szabálytisztelő	introvertált, feladatorientált, a csoport agya, megoldásorientált	pontos, analitikus, maximalista, bizalmatlan

2. táblázat. A DISC viselkedési stílusok rövid leírása



1. ábra. A feltételezett szinergiahálózat a DISC viselkedési stílusok között

A különböző DISC viselkedési stílusok megoszlása egy csoportban erősen befolyásolhatja a csoport morálját és teljesítményét [32], ugyanakkor egy másik tanulmány [3] szerint nincs kapcsolat az egyes típusok és a munkavállalók közötti szinergiák között. A szakirodalom [47] alapján megbecsültük a DISC viselkedési stílusok közötti pozitív és negatív szinergiákat egy kiegyensúlyozott DISC csoportban (lásd *1. ábra*).

A szinergiahálózat pozitív páros szinergiákat tartalmaz D és I, D és C, I és S, valamint S és C közötti elsődleges viselkedési stílusokkal rendelkező egyének között, és negatív páros szinergiákat D és S, valamint I és C között. A részleteket a *3. táblázat* tartalmazza.

Pozitív szinergiájú viselkedésistílus-párok (a legpozitívabb dőlttel jelölve)

Első stílus	Második stílus	Előny az első stílusból	Előny a második stílusból
D	I	vezetés és irányítás	inspiráció és motiváció
D	C	*fókuszálás az eredmények elérésére*	*figyelem a részletekre*
S	I	stabilitás és következetesség	kapcsolatok építése
S	C	együttműködés	figyelem a részletekre

Viselkedésistílus-párok negatív szinergiával

Első stílus	Második stílus	Az első stílus kihívásai	A második stílus kihívásai
D	S	delegálás és nyomásgyakorlás	rugalmatlanság
C	I	távolságtartás	csapongás

3. táblázat. Viselkedésistílus-párosítások szinergiája

A kiválasztás során ügyelnünk kell arra, hogy milyen csoportszerepet milyen viselkedési stílusú embernek szánunk, mert ez erősítheti vagy gyengítheti a viselkedési stílusok közötti pozitív, vagy negatív szinergiapotenciált. A kiválasztás során továbbá figyelembe kell vennünk az egyéni készségeket is, amelyek lehetnek puha készségek, valamint kemény, készség-kapacitással jellemezhető tényezők is.

A tanulmányunk az eddigi szakirodalmi módszerekhez képest két jelentős tudományos és szakmai hozzájárulást fogalmaz meg.

J$_1$ A javasolt modell lehetővé teszi a szakemberek és kutatók számára, hogy rugalmas projektekben modellezzék a puha és kemény készségeket.

J$_2$ A javasolt modell megmutatja, hogyan vizsgálható az autonóm csoportszerveződés és az irányított csoportszerep-kiválasztás hatása.

Az SSPSP javasolt módosítása megmutatja, hogyan modellezhetünk egy agilis szinergiaalapú szoftverprojekt-ütemezési problémát (J_1), ahol

1. végrehajtási prioritások rendelhetők a feladatokhoz;
2. rugalmas függőségek alapján különböző végrehajtási sorrend határozható meg;
3. modellezhetők a munkavállalók közötti szinergiák és csoportszerepek;
4. elkülöníthetők a puha készségek, valamint a kemény készségkapacitások.

Az eredeti SPSP kiterjesztéseinek kezelésére mátrixalapú ábrázolást alkalmaztunk. A mátrixalapú megközelítések előnye, hogy hatékonyan kezelhetjük velük a végrehajtási prioritásokat és a rákövetkezési relációkat [25]. Ha például egy kevésbé fontos feladatot kihagyunk vagy elhalasztunk a projektből, a mátrixot úgy redukáljuk, hogy a kihagyott vagy elhalasztott feladat(ok) megfelelő oszlopát és sorát eltávolítjuk [22]. A mátrixnak ez a redukálása hatékonyan csökkenti az adott problémát. A mátrixalapú tervezés lehetővé teszi több szempont (pl. munkavállalói szinergia, készségek, feladathozzárendelések, viselkedési stílusok) egyidejű ábrázolását (lásd: (J_1 hozzájárulás), amelyeket ún. részmátrixokba (tartományokba, angolul: domain) szervezhetünk [10]. A csoportszerepek meghatározása után vizsgálható a csoportszerepek kiválasztási mechanizmusainak hatása (J_2), és megválaszolható a következő kutatási kérdés (K).

K: A korlátok figyelembevételével, melyik csoportszerep-kiválasztási mechanizmus biztosítja a legrövidebb projektidőtartamot, illetve a legkisebb projektköltséget?

A tanulmány további része a következőképpen épül fel. A 2. fejezet bemutatja a módosított Synergy Mapping Model (SMM) mátrixot és a kiterjesztett SSPSP feladatot. A 3. fejezet ismerteti, hogyan alkalmazzuk ezt a módszert a szabványos, több szakképzettséget igénylő erőforrás-korlátos projektütemezési problémára. A 4. fejezetben mutatjuk be a valós esettanulmányt, ahol láthatjuk, hogyan lehet figyelembe venni a módosított SSPSP feladatban a bevezetett új jellemzőket, például a szinergiákat és a kemény készség-kapacitásokat. Az 5. fejezet az eredményeket és azok értelmezését tartalmazza, végül a tanulmányt a 6. fejezet zárja az eredmények összegzésével és következtetések megfogalmazásával.

2 Alkalmazott módszerek

Az SSPSP problémát egy Multidomain Mapping (MDM) modellben mutatjuk be (Kosztyán és mtsai. [23]), amely egy hat részből álló mátrixmodell. Az SMM mátrix jelen tanulmányban javasolt változata hat *tartományt* és két oszlopvektort tartalmaz. A módosított SMM mátrix egy $(m+n) \times (m+s+n+1)$ mátrix, ahol a munkavállalók száma m, a készségek és/vagy kapacitások száma s, a feladatok száma pedig n. A módosított SMM mátrixot a *2. ábra* szemlélteti, amelyet az alábbiakban részletezünk.

1. Az első tartomány ($\mathbf{Y}$) egy $m \times m$ részmátrix, amely a munkavállalók közötti szinergiát (kapcsolatokat) tartalmazza. Tetszőleges $i \neq j$ esetén

   - $[\mathbf{Y}]_{ij} > 1$ pozitív (vagy kedvező) szinergiát jelent,
   - $[\mathbf{Y}]_{ij} = 1$ semleges szinergiát jelent,
   - $0 < [\mathbf{Y}]_{ij} < 1$ negatív (vagy kedvezőtlen) szinergiát jelent az i és j munkavállalók között.

   Továbbá feltételezzük, hogy a szinergiakapcsolatok mindig szimmetrikusak ($[\mathbf{Y}]_{ij} = [\mathbf{Y}]_{ji}$), az önmagával vett szinergikus kapcsolat pedig semleges: $[\mathbf{Y}]_{ii} = 1$.

   A pozitív (negatív) szinergia azt mutatja meg, hogy két munkavállaló közösen mennyivel hamarabb (később) tud egy feladatot megoldani ahhoz képest, ha a feladatot külön oldanák meg. Fontos megjegyezni, hogy a szinergia nem azonos a szociális kapcsolaterőssséggel, hanem az együttműködés hatékonyságát méri. A szinergia hatását az időtartamra lásd a (4) formula esetében.

2. A második tartomány a készségtartomány ($\mathbf{S}$). A készségtartomány egy $m \times s$ részmátrix, ahol minden készség és/vagy kapacitás nemnegatív szám ($[\mathbf{S}]_{ij} \in \mathbb{R}_0^+$). $s_j := s_{\cdot j} := \left[[\mathbf{S}]_{1j}, [\mathbf{S}]_{2j}, \ldots, [\mathbf{S}]_{mj}\right]$ készségvektorokat jelentenek. Feltételezzük, hogy h *kemény* készség és/vagy kapacitás van, amelyeket az $[s_1, s_2, \ldots, s_h]$ keménykészség-vektor jelöl, és van $s - h$ *puha* készség, amelyeket az $[s_{h+1}, \ldots, s_s]$ készségvektor képvisel. $[\mathbf{S}]_{ij} = 0$ azt jelenti, hogy a i munkavállalónak nincs j készsége. $\mathbf{S}$ tárolhat *bináris* (dichotóm), *sorrendi* vagy *kardinális* mérési szintű készségeket és/vagy kapacitásokat. Bináris ábrázolás esetén $[\mathbf{S}]_{ij} \in \{0, 1\}$. Ordinális skála használata esetén, ha a készségek szintjeit (pl. nyelvi készség szintje, kommunikációs készség szintje stb.) reprezentáljuk, akkor $[\mathbf{S}]_{ij} \in \mathbb{N}$. Ezeket a bináris és ordinális készségeket *nemadditív*nak nevezzük, más szóval a $\sum_i [\mathbf{S}]_{ij}$ összegnek nincs semmi jelentése. Ezzel szemben feltételezzük, hogy a kardinális készségek és/vagy kapacitások (általában a kemény készségek kapacitása) *additívak*, és $[\mathbf{S}]_{ij} \in \mathbb{R}_0^+$. A szinergikus tényezők is módosíthatják a készségeket és/vagy kapacitásokat, ezért mi a következő modellt javasoljuk. Legyen ε a munkavállalók egy részhalmaza, ekkor ε *együttes* j-edik *készség-kapacitása* a következőképpen számítható:

$$S_j^{\varepsilon} := \overline{Y}_{\varepsilon} \cdot \sum_{i \in \varepsilon} [\mathbf{S}]_{ij} \tag{1}$$

ahol $\overline{Y}_{\varepsilon}$ a szinergiák *mértani átlaga*:

$$\overline{Y}_{\varepsilon} := \begin{cases} 1 & \text{ha } |\varepsilon| \leq 1 \\ \sqrt[\eta]{\prod\limits_{i,j \in \varepsilon,\, i<j} [\mathbf{Y}]_{i,j}} \quad \text{ahol } \eta = \frac{|\varepsilon| \cdot (|\varepsilon| - 1)}{2} & \text{ha } |\varepsilon| > 1 \end{cases} \tag{2}$$

3. A harmadik tartomány egy munkaerő-hozzárendelési tartomány ($\mathbf{M}$). $\mathbf{M}$ egy $m \times n$-es részmátrix, ahol $[\mathbf{M}]_{ij} \in [0,1]$ az i-edik munkavállaló j-edik feladathoz való hozzárendelésének *maximális* relatív mennyiségét jelenti. Ha $[\mathbf{M}]_{ij} = 0$ ($[\mathbf{M}]_{ij} = 1$), akkor i-edik alkalmazott nincs hozzárendelve (teljes mértékben hozzá van rendelve) a j-edik feladathoz.

4. A negyedik tartomány a tevékenység- vagy feladattartomány ($\mathbf{A}$), egy $n \times n$-es mátrix, $[\mathbf{A}]_{ij} \in [0,1]$. Az átló a feladatok (tevékenységek) elvégzésének relatív prioritását jelöli: $[\mathbf{A}]_{ii} = 1$ *kötelező* feladatot jelöl, amelyet mindenképpen el kell végezni. Ezek a tevékenységek nem hagyhatók el, és nem sorolhatók át későbbi részprojektbe. $0 < [\mathbf{A}]_{ii} < 1$ az ún. *kiegészítő* feladatokat jelölik, amelyek a korlátozásoktól függően elhagyhatók vagy elhalaszthatók. A magasabb $[\mathbf{A}]_{ii}$ érték magasabb prioritást (nagyobb pontszámot) jelent. Fontos megjegyezni, hogy egy elhalasztott feladat függőségi kapcsolatai és idő-, költség- és erőforrásigényei is elhalasztódnak.
$[\mathbf{A}]_{ij} (i \neq j)$ az a_i és a_j feladatok közötti *rákövetkezési* relációkat jelölik. Ha a_i-nek véget kell érnie, mielőtt a_j elkezdődik, akkor az a_i és a_j feladat közötti *fix* függőségről beszélünk, ezt $[\mathbf{A}]_{ij} = 1$ illetve $a_i \prec a_j$ jelöli (soros végrehajtás). Ha *semmilyen* precedencia nincs a_i és a_j között, akkor $[\mathbf{A}]_{ij} = 0$ illetve $a_i \sim a_j$ (párhuzamos végrehajtás). Végül, $0 < [\mathbf{A}]_{ij} < 1$ ($a_i \bowtie a_j$) a *rugalmas* függőségeket jelentik, amelyek a korlátoktól függően jelölhetnek soros vagy párhuzamos végrehajtást, ezt az optimalizáló algoritmusnak kell eldöntenie.
Fontos, hogy az optimalizálás után a korlátoktól függően minden $0 < [\mathbf{A}]_{ii} < 1$ és $0 < [\mathbf{A}]_{ij} < 1$ $(i \neq j)$ értéknek vagy 1-re vagy 0-ra kell változnia. Ez azt jelenti, hogy a javasolt algoritmusnak el kell döntenie, hogy mely kiegészítő feladatokat kell befejezni vagy elhalasztani, és mely rugalmas kapcsolatokat kell előírni vagy feloldani.

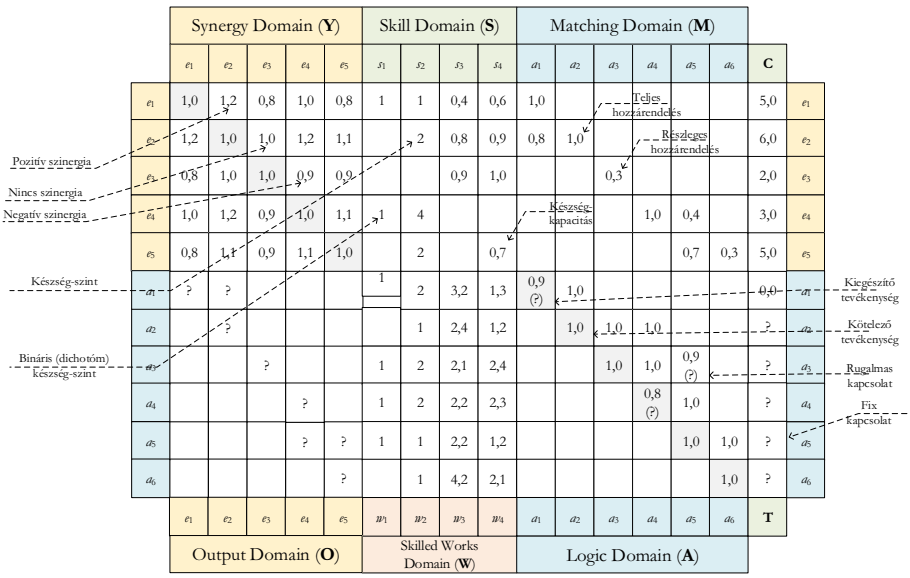
5. Az ötödik tartomány az előírt munkatartalom részmátrix ($\mathbf{W}$) egy $n \times s$ méretű mátrix, ahol $[\mathbf{W}_{ji}]$ a j feladathoz szükséges minimális i készség (kapacitást) jelöli. A bináris és ordinális készségek esetében $[\mathbf{W}_{ji}]$ a j feladat elvégzéséhez szükséges készségek minimális követelményét (például a kommunikáció vagy a nyelvtudás minimális szintjét), míg a készség-kapacitás esetében $[\mathbf{W}_{ji}]$ a készség-kapacitások minimális mennyiségét (például tesztelt függvények és dokumentált programozási kódok minimális értékét) jelenti.

6. Az utolsó tartomány a kimeneti tartomány ($\mathbf{O}$), amely az SSPSP algoritmus megoldását tartalmazza. $\mathbf{O}$ egy $n \times m$-es, nemnegatív valós számokból álló mátrix, ahol az $[\mathbf{O}]_{ji} > 0$ elem az e_i alkalmazott a_j feladathoz való (tényleges) hozzárendelését jelenti. Az SSPSP probléma megoldásakor az algoritmus eldönti, hogy az e_i munkavállaló munkaidejének *ténylegesen* mekkora *hányadát* tölti az a_j feladat megoldásával, ezt az $\mathbf{O} \in I\!R^{n \times m}$ mátrix (output domain) $\mathbf{O}_{ji}$ eleme adja meg. A $0 \leq \mathbf{O}_{ji} \leq \mathbf{M}_{ij}$ és $\sum_{j=1}^{n} [\mathbf{O}]_{ji} \leq 1$ egyenlőtlenségeknek nyilván minden

$j = 1, 2, \ldots, n$ és $i = 1, \ldots, m$ esetében teljesülniük kell.

7-8. A módosított SMM mátrix két extra oszlopvektort is tartalmaz. $\mathbf{C}$, az első egy $m \times 1$ oszlopvektor, amely a munkavállalók fizetését tartalmazza, a második, $\mathbf{T}$ pedig egy $n \times 1$-es oszlopvektor, amely a feladatok tervezett kezdési időpontját tartalmazza.

A 2. ábra egy kitöltött módosított SMM mátrix példáját mutatja, ahol a munkavállalók száma öt, a feladatok száma pedig hat (négy kötelező feladat, két kiegészítő feladat található). A példa egy bináris, egy ordinális készséget és két készség-kapacitást ábrázol. A „?” szimbólum jelöli azokat a változó cellákat, amelyeket a javasolt algoritmussal optimalizálni kell.



2. ábra. A javasolt, módosított SMM-mátrix

Az a_j tevékenység *időtartamát* $a_j^{dur}(\mathbf{O})$ jelöli. (Ez az (5) és (6) egyenletekben kiszámított, szinergiafaktorral módosított erőforrásoktól függ). a_j *kezdési ideje* $a_j^{start}(\mathbf{O})$, befejezési ideje $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + a_j^{dur}(\mathbf{O})$ (lásd a (7) egyenletet). A projekt *átfutási idejét* p_{dur} (Total Project Time (TPT)), költségét pedig p_{cost} (Total Project Time (TPC)) jelöli. Az e_i munkavállaló havi *bérét* e_i^{salary} vagy $[\mathbf{C}]_i$ jelöli.

Mivel az a_j feladat $[\mathbf{W}]_{jk}$ szakképzett munkát igényel, az a_j feladat s_k követelményének (készségének) teljesítéséhez *szükséges időtartam*, szinergiák figyelembevétele nélkül:

$$a_{jk}^{dur}(\mathbf{O}) = \frac{[\mathbf{W}]_{jk}}{\sum_{i=1}^{m}([\mathbf{S}]_{ik} \cdot [\mathbf{O}]_{ji})}, \tag{3}$$

míg a szinergiák figyelembevételével *kiigazított szükséges időtartam*:

$$a_{jk}^{dur,adj}(\mathbf{O}) = \frac{[\mathbf{W}]_{jk}}{\overline{Y}_{\varepsilon_j} \cdot \sum_{i=1}^{m}([\mathbf{S}]_{ik} \cdot [\mathbf{O}]_{ji})}\,, \tag{4}$$

ahol $\varepsilon_j := \{i : 0 < [\mathbf{O}]_{ji}\}$ az a_j feladat elvégzésére kiválasztott munkavállalók összessége.

Vegyük észre, hogy amennyiben $\overline{Y}_{\varepsilon_j} > 1$, vagyis pozitív szinergiáról beszélünk, akkor az időtartam csökken. Míg $0 < \overline{Y}_{\varepsilon_j} < 1$ esetén a kiigazított időtartam növekszik az eredetileg becsülthöz ($a_{jk}^{dur}(\mathbf{O})$) képest.

A szinergiák figyelembevétele nélkül az a_j feladat időtartama:

$$a_j^{dur}(\mathbf{O}) = \max_{0<[\mathbf{W}]_{jk}} \{a_{jk}^{dur}(\mathbf{O})\}\,. \tag{5}$$

Szinergiák figyelembevételével ez az érték:

$$\widetilde{a}_j^{dur}(\mathbf{O}) := a_j^{dur,adj}(\mathbf{O}) = \max_{0<[\mathbf{W}]_{jk}} \{a_{jk}^{dur,adj}(\mathbf{O})\}\,. \tag{6}$$

Az időtartamokat a tevékenységek befejezési idejének kiszámításához használjuk:

$$a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + \widetilde{a}_j^{dur}(\mathbf{O})\,, \tag{7}$$

ahol

$$a_j^{start}(\mathbf{O}) \geq \begin{cases} 0 & \text{ha } \nexists a_i \in A : a_i \prec a_j \\ \max\{a_i^{end}(\mathbf{O}) : a_i \prec a_j\} & \text{különben}\,. \end{cases} \tag{8}$$

Az a_i tevékenység akkor *kritikus*, ha sem a kezdeti, sem a befejezési ideje nem változtatható meg a projektidő optimalizálásakor, csúszása a projekt csúszását eredményezné.

A fentiekben kiszámított értékek alapján a projekt időtartamát a következőképpen tudjuk kiszámítani:

$$\mathrm{TPT}_{nosyn} := \max\{a_j^{end}(\mathbf{O}) : j = 1, \ldots, n\} \tag{9}$$

$$\mathrm{TPT}_{syn} := \max\{\widetilde{a}_j^{end}(\mathbf{O}) : j = 1, \ldots, n\}. \tag{10}$$

Ha minden kiegészítő feladatot elhalasztunk, és minden rugalmas feladatot kizárunk (formálisan: $\mathbf{A}_{\min} = \lfloor \mathbf{A} \rfloor$); teljes feladatkiosztás esetén (formálisan $\mathbf{O} = \mathbf{M}^T$) minimális TPT-t ($\mathrm{TPT}_{\min}$) kapunk. A gyakorlatban azonban ez a megközelítés a korlátok miatt általában nem megvalósítható. A maximális TPT végtelen, ha nincs hozzárendelés.

Ha a projekt költségének (TPC) kiszámításánál, az egyszerűsítés miatt csak az emberi erőforrás költségét tekintjük, akkor ez nem lesz más, mint a projektben munkájukért fizetett munkavállalók bérének összege. Mivel a

pozitív szinergia csökkenti, a negatív szinergia pedig növeli az a_j^{dur} időtartamot $\widetilde{a}_j^{dur}$-ra, a projekt költsége kiszámítható szinergikus hatással és anélkül. Így megkapjuk a TPC_{syn} és a TPC_{nosyn} értékeket. Formálisan:

$$\mathrm{TPC}_{syn} = \sum_{i=1}^{m}\sum_{j=1}^{n}([\mathbf{C}]_i \cdot [\mathbf{O}]_{ji} \cdot \widetilde{a}_j^{dur}(\mathbf{O})), \tag{11}$$

$$\mathrm{TPC}_{nosyn} := \sum_{i=1}^{m}\sum_{j=1}^{n}([\mathbf{C}]_i \cdot [\mathbf{O}]_{ji} \cdot a_j^{dur}(\mathbf{O})). \tag{12}$$

A költségek maximális összege $\mathrm{TPC}_{\max}$ maximális hozzárendelés esetén, minden kiegészítő feladat elvégzése mellett fordul elő. A minimális érték 0, ha egyik feladathoz sincs erőforrás-hozzárendelés.

A Total Project Score (TPS)-t nem befolyásolják a szinergiák, ez csak a teljesítendő feladatok halmazától függ, amelyet $\mathcal{A}$ jelöl:

$$\mathrm{TPS} := \sum_{i\in\mathcal{A}}[\mathbf{A}]_{ii} \tag{13}$$

$\mathrm{TPS}_{\min}$ ($\mathrm{TPS}_{\max}$) akkor következik be, ha az összes kiegészítő feladatot elhalasztják (megvalósítják).

Ily módon a fentebb leírt matematikai modell az SSPSP-hez képest képes kezelni a kemény és puha készségszinteket is rugalmas környezetben, és képessé válik az autonóm csoportszerveződés és az irányított csoportszerep-kiválasztás hatásának vizsgálatára.

2.1 Lehetséges célfüggvények

A következő célfüggvényeket vehetjük figyelembe az optimalizálás során:

$$\mathrm{TPT} \rightarrow \min, \tag{14}$$

$$\mathrm{TPC} \rightarrow \min, \tag{15}$$

$$\mathrm{TPS} \rightarrow \max. \tag{16}$$

A fenti célfüggvényeket egy többcélú optimalizálási feladat felírására is lehet alkalmazni, mi azonban ehelyett negyedik célfüggvénynek a három célfüggvény kompozit mutatóját vettük figyelembe:

$$z := 1 - \sqrt[3]{\frac{C_t - \mathrm{TPT}}{C_t - \mathrm{TPT}_{\min}} \cdot \frac{C_c - \mathrm{TPC}}{\mathrm{TPC}_{\max}} \cdot \frac{\mathrm{TPS} - C_s}{\mathrm{TPS}_{\max} - C_s}} \rightarrow \min. \tag{17}$$

Ebben a tanulmányban a négy célfüggvényt külön-külön is vizsgáltuk, melyek a legrövidebb átfutási időt (min TPT), legkisebb költséget (min TPC), legnagyobb tartalmat megadó pontérték (max TPS), valamint ezek kombinációi lehettek.

2.2 Korlátok

A szakirodalom az SPSP feladatra számos korlátot fogalmaz meg, ugyanakkor a mi esetünkben a tevékenységek végrehajtási struktúrája rugalmas, itt megkülönböztetünk puha és kemény készségeket, és a hozzárendelés sem bináris, hanem százalékos, ezért a korlátok a szakirodalmi korlátokhoz képest némiképp módosulnak. A korlátok a [14, 21, 23, 33, 53] szakirodalmak és a feladat általánosítása miatt a következőképpen írhatók fel:

CR$_1$ A projekthez szükséges összes erőforrás emberi erőforrás, és az mindig rendelkezésre áll. Az egyes e_i munkavállalók foglalkoztatása a *projektben* nem haladhatja meg az előre meghatározott *maximum* értékét: $e_i^w := \sum_{j=1}^{n}[\mathbf{O}]_{ji} \leq e_i^{maxw} := \sum_{j=1}^{n}[\mathbf{M}]_{ij}$. Nyilvánvaló, hogy $0 \leq e_i^w \leq 1$ a $\sum_{j=1}^{n}[\mathbf{O}]_{ji} \leq 1$ feltétel szerint. Ezenkívül minden egyes tevékenységet legalább egy emberi erőforrásnak kell végeznie.

CR$_2$ A tevékenységhez szükséges készségek halmazának (S_{necc}) az adott tevékenységet végző munkavállalók készségeinek összességét tartalmazó halmaz (S_{empl}) részhalmazának kell lennie ($S_{necc} \subseteq S_{empl}$). Ezen túlmenően az érintett munkavállalók "közös" készségszintje legalább a tervezett tevékenység által megkövetelt szintnek feleljen meg. Tehát *nemadditív* készségek esetén bármely i, j esetén $[\mathbf{S}]_{\ell j} \geq [\mathbf{W}]_{ij}$ valamely ℓ esetén, amelyre e_ℓ az a_i-hez van rendelve, *additív* készségek esetén pedig $S_j^\varepsilon \geq [\mathbf{W}]_{ij}$ minden j -re, az a_i-hez rendelt ε munkavállalók halmazára. Végül, minden egyes a_i kritikus tevékenység és s_j puha készség esetében, amely legalább w_{ij} szinten szükséges a_i-hez, legalább egy e_ℓ munkavállalót kell beosztani a_i-hez, aki rendelkezik az s_j készséggel a w_{ij} minimális szinten, azaz $s_{\ell j} \geq w_{ij}$.

Az SSPSP-ben további megkötések vannak a rugalmas projektek kezelésére és a végrehajtott feladatok halmazának meghatározására.

CR$_3$ A TPS-nek nagyobbnak kell lennie, mint a meghatározott projekttartamra (project scope) vonatkozó korlátozás C_s, formálisan: TPS$> C_s$.

A további hat megkötés a projekt tervezésénél fontos.

CR$_4$ A túlmunka egy bizonyos szintig megengedett ($E^w = \sum_{i=1}^{m} e_i^w \leq K^w$ valamilyen konstans K^w esetén, kisebb kivételekkel).

CR$_5$ A fejlesztési költségek tartalmazzák a projektekhez rendelt összes munkavállaló teljes bérét, és TPC nem lehet nagyobb, mint a C_c költségkorlát.

CR$_6$ A projekt TPT átfutási ideje nem lehet nagyobb, mint a C_t időkorlát.

CR$_7$ Minden feladatot egyszer kell elindítani és megszakítás nélkül végrehajtani.

CR$_8$ Az egyes tevékenységek különböző készségeihez rendelt összes munkavállaló egyidejűleg kezdje meg munkáját.

CR$_9$ A személyzet a teljes fejlesztési folyamat során rögzített.

Fontos megjegyezni, hogy a szakirodalomban általában kikötik, hogy az előre meghatározott rákövetkezési relációkat meg kell tartani. Egy rugalmas projektben azonban a végső végrehajtási precedencia változhat.

2.3 Az alkalmazott hibrid genetikus algoritmus

Mivel az SPSP probléma már önmagában is NP-nehéz, melyet Xiao és mtsai. [58] igazoltak, ezért az eredeti SSPSP, ami az SPSP kiterjesztése, szintén NP-nehéz feladat [23]. Kosztyán és mtsai. [23] egy hibrid genetikus algoritmust javasoltak az SSPSP probléma megoldására. A javasolt hibrid genetikus algoritmus (angolul: Hybrid Genetic Algorithm (HGA)) két fázisból áll. Az első fázisban egy genetikus algoritmust (Genetic Algorithm (GA)) használnak az elvégzendő feladatok és feladat-hozzárendelések meghatározására. A második fázisban egy Nelder-Mead Minimization (NMM) módszert használnak az ütemezett kezdések (Scheduled Start Time (SST)) finomítására az erőforrásigények kiegyensúlyozása érdekében. Kibővítettük és párhuzamosítottuk ezt az algoritmust, amelyet MATLAB-ban is megvalósítottunk. A HGA eredeti beállításai több cikkben [23, 24, 26] megjelentek korábban, ezért jelen tanulmányban csak a kiterjesztésekre és módosításokra helyezzük a hangsúlyt. A tanulmány eredeti gondolata Kosztyán és mtsai. [24] cikkben jelent meg. Ennek jelentősen kibővített, a gyakorlati alkalmazhatóságra sokkal jobban fókuszáló változatát olvashatja az olvasó.

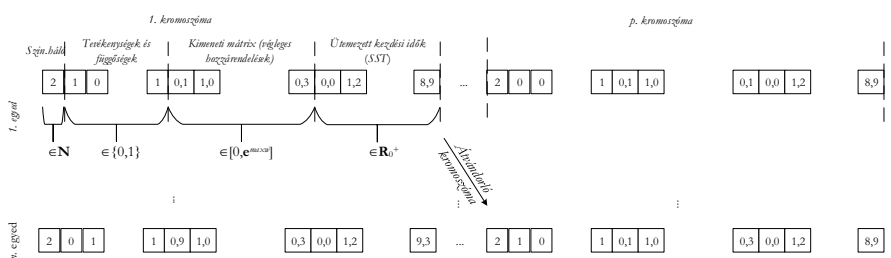
Egy kromoszóma a módosított SSPSP egy lehetséges megoldását kódolja. Az eredeti kromoszómaszerkezet javasolt módosítása megkülönbözteti a készségeket bináris és ordinális készségekre, amelyek általában puha készségek, és additív készségkapacitásokra, amelyek általában a kemény készségek kapacitását írják le. Végül a kromoszómákat multikromoszóma struktúrába rendeztük (lásd a *3. ábrát*), hogy csökkentsük a számítási időt, és megosszuk a legjobb kromoszómákat a párhuzamos számításokban. A javasolt kromoszómaszerkezet négy részből áll.

A kromoszóma első eleme a szinergiahálózat kiválasztott száma. Ez egy 1 és N közötti sorszám. A második rész a kiegészítő feladatok és a rugalmas függőségek elvégzésének döntési eredményének bináris sorozata. A kromoszóma ezen részének hossza $n_F = n_s + n_f$, ahol n_s a kiegészítő feladatok száma, n_f pedig a rugalmas függőségek száma. A kromoszóma harmadik része kódolja a munkaerő-hozzárendelési arányokat. Ezeknek a valós értékeknek a [0,1] intervallumban kell maradniuk. A hozzárendelési arányok száma (n_A), amely a maximális munkaerő-hozzárendelési részmátrix (**M**) nullától eltérő elemeinek száma. Az utolsó rész a feladatok ütemezett kezdését (SST) kódolja. Ebben a részben az elemek száma n. Ezért egy kromoszómavektor elemeinek száma $N_c = 1 + n_F + n_A + n$. A kromoszóma mind a négy része más

típusú; ezért ezekhez a részekhez különböző keresztezési, mutációs és szelekciós mechanizmusokat kell javasolni. A kromoszóma első két részében egységes keresztezési mechanizmust alkalmazunk. A szülők azonban lehet, hogy nem megengedett megoldást kódolnak; ezért feltételezzük, hogy a megengedett megoldást kódoló szülők génjei tízszer dominánsabbak. Más szóval, egy gén tízszer nagyobb valószínűséggel származik megengedett megoldást kódoló szülőktől, mint nem megengedett megoldást kódoló szülőktől. A kromoszóma harmadik és negyedik (folytonos értékeket kezelő) részéhez aritmetikai keresztezési operátort használunk. Mivel minden kromoszóma kódolja a módosított SSPSP egy-egy leképzését, a megoldás megengedhetősége ellenőrizhető. Bár a megengedhetőség kromoszómánkénti vizsgálata időt vesz igénybe, eredményeink azt mutatják, hogy a megengedett megoldásokat kódoló egyedek számának növelése jobb konvergenciát eredményez anélkül, hogy legjobb megvalósítási értékektől jelentősen távolodnánk. A genetikus algoritmus esetében hangoltuk a paramétereket is (melyet angolul: Design of Experiments (DoE)-nek neveznek), ahol a hangolás során ezt a paramétert is megvizsgáltuk (lásd a 2.4 fejezetet).

A mutáció során kétlépcsős mutációs eljárást alkalmaztunk, amelyben az első lépés általános, így a kromoszóma minden részére végrehajtjuk. Az első lépésben az algoritmus kiválasztja a kromoszóma, mint vektor génjeinek (mint elemeinek) egy részét mutációra, ahol minden gén mutációjának a valószínűsége egy egyenletes eloszlás szerint generált érték lesz, ha ez az érték nagyobb, mint p, akkor a gén mutálódhat. A második lépésben, megkülönböztetjük a bináris, diszkrét és folytonos elemeket, melyekre eltérő mutációs mechanizmust választottunk. A valós részekhez véletlen számot adunk hozzá a felvehető korlátokon belül, míg a bináris biteket átbillentjük.

A számítási idő csökkentése érdekében egy multikromoszóma-szerkezetet határoztunk meg. Mivel a kromoszómák hossza minden futásban egyenlő, a GA párhuzamosítható. Egy többkromoszómás szerkezetben P kromoszómavektorok vannak. A valódi párhuzamos futtatás érdekében P értéke általában a processzorok (vagy grafikus társprocesszorok) számának többszöröse. A készlet M multikromoszómát tartalmaz egy generáción belül. A processzorok párhuzamosan értékelik őket, és a kiválasztott kromoszómák vagy kromoszómák részei vándorolhatnak egymás között (lásd a *3. ábrát*).



3. ábra. A javasolt multikromoszóma-struktúra

2.4 A genetikus algoritmus hiperparamétereinek hangolása

Mivel Kosztyán és mtsai. [23] is genetikus algoritmussal oldották meg az eredeti SSPSP problémát, az ott alkalmazott paramétereket használtuk a kiterjesztésnél is. A DoE módszerrel a GA hiperparamétereit Reeves és Wright [46] tanulmányát követve állítottuk be, az optimalizálási eredményeket pedig egy regressziós fa (a MATLAB regressziós tanuló alkalmazása) segítségével értékeltük. A számítás során 10-szeres keresztellenőrzést használtunk, és a hiperparamétereket Bayes-optimalizálással hangoltuk. A célfüggvény a konvergenciasebesség volt. Hangolás után a GA következő paramétereit használtuk a későbbiekben (lásd a *4. táblázatot*).

Operátorok	Paraméterek
Populáció mérete	250
Kromoszómák csoportjai	4
Körmérkőzéses kiválasztás mérete	9
Elit egyedek száma	0,05
Keresztezési arány	0,82
Megengedett megoldást kódoló kromoszómák aránya a keresztezésben	0,88
Egy gén mutációjának valószínűsége egy kromoszómában	0,05
A migrált kromoszómák maximális aránya	0,09
Tűréshatár	10^{-8}
Maximális iteráció	150

4. táblázat. A GA hangolt hiperparaméterei

A GA hiperparamétereinek hangolása után a négy kromoszómacsoportban lévő populációk száma 250 lett. A legjobb kiválasztási mechanizmus a körverseny volt, ahol a körverseny szereplőinek száma 9 volt. Az elit egyedek száma 5% volt, ami azt jelenti, hogy a jelenlegi generációból hány egyed marad biztosan fenn a következő generációban is. A keresztezési arány megadja az egyes populációk (az elit egyedek kivételével) tagjaiból keresztezett utódok hányadát. A megengedett megoldást kódoló kromoszómák aránya a keresztezésben 88% volt. Ez az érték gyors konvergenciát biztosít, de a nem megengedett egyedek segítenek abban, hogy ne csak lokális minimumokra találjunk az optimalizálás során. A génmutáció valószínűsége 5% volt, míg a kromoszómavándorlás maximális aránya 9% volt. Az alkalmazott tolerancia 10^{-8} volt, míg 150 iteráció mindig elegendő volt.

3 A módszer alkalmazása meglévő tesztadatbázison

A módosított SSPSP algoritmus elsődleges teszteléséhez a Myszkowski et al. [38] által létrehozott imopse adatbázist használtuk. Ebből az adatbázisból a 15_6_10_9 projektet használtuk fel, mely 15 feladatot, 6 munkavállalót és 9 készséget tartalmaz. Ez az adatbázis azonban olyan feladatokat tartalmaz, amelyek nem veszik figyelembe a következőket:

- *szinergiák*; ezért először a szinergiatartományt egy egyesekből álló mátrixnak tekintettük ($\mathbf{Y} := \mathbf{1}, [\mathbf{Y}]_{ij} := 1$);
- *rugalmas függőségek*; ezért a $\mathbf{A}$ részmátrixban csak bináris értékeket használtunk;
- *készségteljesítmények*; ezért a készségeket ordinális változóknak tekintettük;
- *hozzárendelési arány*; ezért csak 0 vagy 1 értéket engedélyeztünk az $\mathbf{M}$-ben és $\mathbf{O}$-ban.

Az *5. táblázat* mutatja a hozzárendelési és függőségi mátrixokat: a munkavállalók készségeit (5a. táblázat), a precedenciamátrixot (5b. táblázat) és a feladat-felelősség mátrixot (5c. táblázat).

a. Készségek

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
e_1					1		2		2
e_2				1	2		1	1	
e_3	2					1	1	1	
e_4	2			2			2		2
e_5			1			2	2		
e_6	1	2							1

b. Logikai végrehajtás (precedenciamátrix)

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
a_1	1									1					
a_2		1	1	1											
a_3			1												
a_4				1						1					
a_5					1										
a_6						1	1								
a_7							1		1				1		
a_8								1							
a_9									1	1			1		
a_{10}										1				1	
a_{11}											1				
a_{12}												1			
a_{13}													1		
a_{14}														1	
a_{15}															1

c. Feladat-felelősség mátrix

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
e_1		1				1						1			
e_2					1			1		1					
e_3	1										1		1		1
e_4			1	1											
e_5									1					1	
e_6							1								

5. táblázat. Hozzárendelési és függőségi-mátrixok

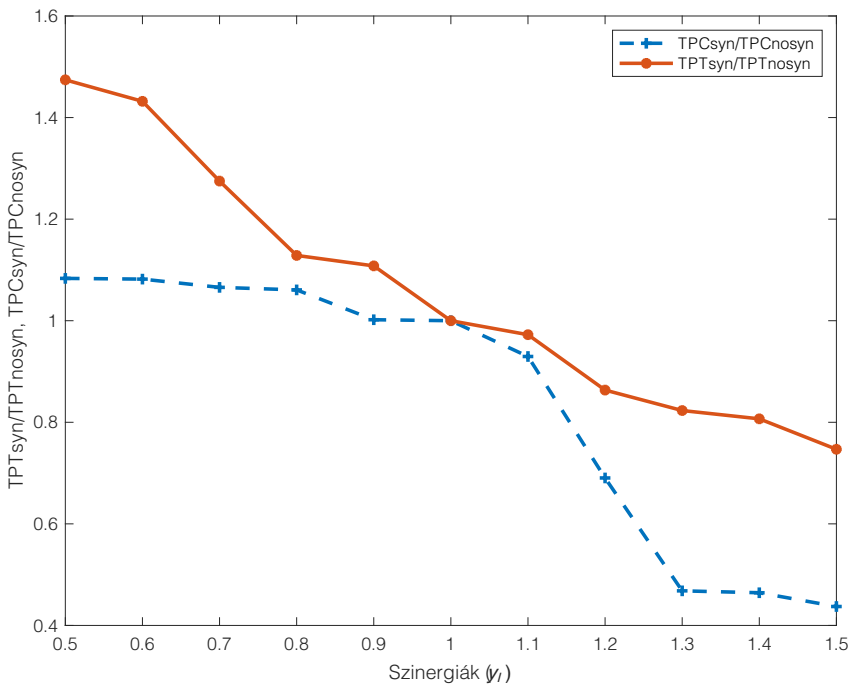
Feladatonként egy fős korlátozás miatt az értékelésünket feladatonként maximum egy résztvevőre korlátoztuk. Az optimalizálás után az időtartamok

$\mathbf{T}$ vektora $\mathbf{T} = [26{,}0006; 21{,}0008; 24{,}0001; 21{,}0004; 39{,}0000; 33{,}0004; 13{,}0014; 33{,}0004; 13{,}0014; 12{,}0000; 9{,}0003, 16{,}0002; 9{,}0001; 36{,}0004; 14{,}0000; 31{,}0000; 17{,}0000]$, ahol optimális hozzárendelések esetén az időtartamok értékei $\mathbf{T}_{opt} = [26; 21; 21; 39; 33; 13; 12; 9; 16; 9; 36; 14; 31; 17]$. A javasolt HGA módszerrel történő megoldás után a TPC 16279,1-nek adódott, míg az optimális TPC 16278,6 volt. Annak ellenére, hogy az eredmények nagyon közel állnak a validált valós értékekhez, a javasolt módszer valódi előnye akkor jelentkezik, amikor a munkavállalók közötti szinergiákat, valamint a megengedett készség-kapacitásokat is figyelembe vesszük (lásd a 4. fejezetet).

Bár az eredeti tesztadatkészlet nem határoz meg szinergiákat a munkavállalók között, a kísérlet második fázisában a következő szinergiatartományokat vettük figyelembe ($\mathbf{Y}_I$).

$$[\mathbf{Y}_I]_{ii} = 1, \quad [\mathbf{Y}_I]_{ij} = y_I, \quad y_I \in \{0{,}5, 0{,}6, \ldots, 1{,}5\}, \quad i \neq j. \tag{18}$$

A *4. ábra* az átfutási idő és a költségek függőségét mutatja a szinergia változásának függvényében. Figyelembe véve a (9)–(10) és a (11)–(12) egyenleteket, TPT és TPC kiszámítása során figyelembe vettük a szinergiákat, a változást pedig a szinergiák nélkül számolt értékekhez viszonyítottuk. Ha a $\mathrm{TPT}_{syn}/\mathrm{TPT}_{nosyn}$ ($\mathrm{TPC}_{syn}/\mathrm{TPC}_{nosyn}$) értéke kisebb, mint egy, akkor látható, hogyan csökken az átfutási idő (teljes költség), míg az egynél nagyobb érték a negatív szinergiák hatását mutatja.



4. ábra. A teljes projektidő és a projekt költségének alakulása a szinergia függvényében

Mivel ebben a feladatban a munkavállalók közötti szinergia egyszerre, azonos irányba változik, az átfutási idők és a költségek is érzékenyek a szinergia változásaira. A 4. ábra azt mutatja, hogy a költségek ($\mathrm{TPC}_{syn}/\mathrm{TPC}_{nosyn}$) érzékenyebbek a pozitív szinergiákra ($y_I > 1$), míg az átfutási idő (lásd: $\mathrm{TPT}_{syn}/\mathrm{TPT}_{nosyn}$) érzékenyebb a negatív szinergiákra ($y_I < 1$).

4 Az esettanulmány bemutatása

A javasolt módszerünk alkalmazását egy multinacionális, csúcstechnológiás autóipari vállalat kutatás-fejlesztési (K+F) projektjén vizsgáltuk meg. A K+F részleg fő tevékenysége a szoftverfejlesztés a vállalat által gyártott fizikai termékekhez, amely szoftverek esetenként akár több millió kódsorból is állhatnak. A szoftverkódokat agilis környezetben, iterációkban fejlesztik, amelyeket részprojekteknek vagy más néven sprinteknek nevezünk. Az esettanulmány során vizsgált projekt a vállalat egy valós szoftverfejlesztési projektének egy iterációja kilenc fő feladattal, ahol egy kényelmi funkció fejlesztésén kívül minden feladat kötelező. Minden egyes funkció megvalósítása és a vevői követelmények alapján való tesztelése külön-külön, de párhuzamosan is megvalósítható. A szoftverrészek integrációját akkor lehet elkezdeni, ha az összes funkciót lefejlesztették és letesztelték, kivéve egy olyan funkciót, amelynek tesztelése az integrációs teszt része. A feladatok elvégzéséhez általánosan 4 fő szükséges, akik részt vettek egy kijelölt DISC tréningen, ahol egy erre kijelölt tréner azonosította a viselkedési stílusukat.

Az alkalmazott DISC módszer alapján négy szinergiahálózatot adhatunk meg, attól függően, hogy melyik viselkedési stílust jelöljük vezetőnek, valamint egy ötödik szinergiahálózat is létezik, amely során a csoport nem választ vezetőt. A csoport négy tagját a DISC 4 különböző viselkedési stílusa szerint választottuk ki: domináns (e_1), befolyásoló (e_2), kitartó (e_3) és szabálytisztelő (e_4). A figyelembe vett hat puha készség: vezetői készség (s_1), kommunikációs készség (s_2), csapatjátékos attitűd (s_3), problémamegoldó készség (s_4), interperszonális készség (s_5) és elemző gondolkodás (s_6). Az adatokat tíz fokozatú Likert-skálával mértük, és elosztottuk őket tízzel, hogy normalizált adatokat kapjunk. A szakirodalom szerint [19, 30, 41] a domináns viselkedési stílusú munkavállalók rendelkeznek a legjobb vezetői készségekkel, a befolyásoló stílusú emberek pedig a legjobb kommunikációs készségekkel és csapatjátékos attitűdökkel (lásd *6. táblázat*). Míg a puha készségek általában a viselkedési stílustól függenek, a kemény készségek kapacitása általában a korábbi tapasztalatoktól függ. Ezek a kemény készségek a következők voltak: egy hét alatt lefejlesztett funkciók száma (s_7), egy hét alatt megtervezett felhasználói interfészek (User Experience/User Interface (UX/UI)) száma (s_8), egy hét alatt ledokumentált fejlesztés és tesztelés száma (s_9), valamint az egy hét alatt elvégzett tesztek száma (s_{10}).

DISC típusok	Puha készségek						Kemény készségek			
	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
e_1 (D)	1,0	0,8	0,3	0,8	0,8	0,8	4,0	1,5	2,0	1,5
e_2 (I)	0,8	1,0	1,0	0,6	1,0	0,4	0,0	3,0	1,5	1,0
e_3 (S)	0,5	0,5	0,9	0,5	0,3	0,6	2,5	2,0	3,5	3,0
e_4 (C)	0,4	0,5	0,5	1,0	0,4	1,0	5,0	0,0	4,0	1,5

6. táblázat. Készségek és a viselkedési stílusok kapcsolata

A *7. táblázat* az egyes feladatokhoz szükséges készségkövetelményeket mutatja be.

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}
a_1	1,00	1,00	1,00	0,35	0,83	0,83	4,00	6,50	9,50	4,00
a_2	0,35	0,38	0,60	0,81	0,63	0,75	10,00	6,00	6,00	4,00
a_3	0,42	0,42	0,80	0,77	0,83	0,83	9,20	3,50	8,00	3,50
a_4	0,38	0,35	0,48	0,96	0,79	0,83	10,50	5,50	5,50	2,50
a_5	0,35	0,38	0,56	0,85	0,63	0,79	9,80	6,00	5,50	3,00
a_6	0,31	0,31	0,40	0,73	0,42	0,88	8,00	6,50	4,00	3,50
a_7	0,81	0,85	0,80	0,38	0,83	1,00	3,50	4,50	2,00	6,00
a_8	0,38	0,77	0,36	1,00	0,42	0,83	10,00	6,00	4,00	4,00
a_9	0,96	0,96	0,40	0,35	1,00	0,17	2,00	1,50	9,50	7,00

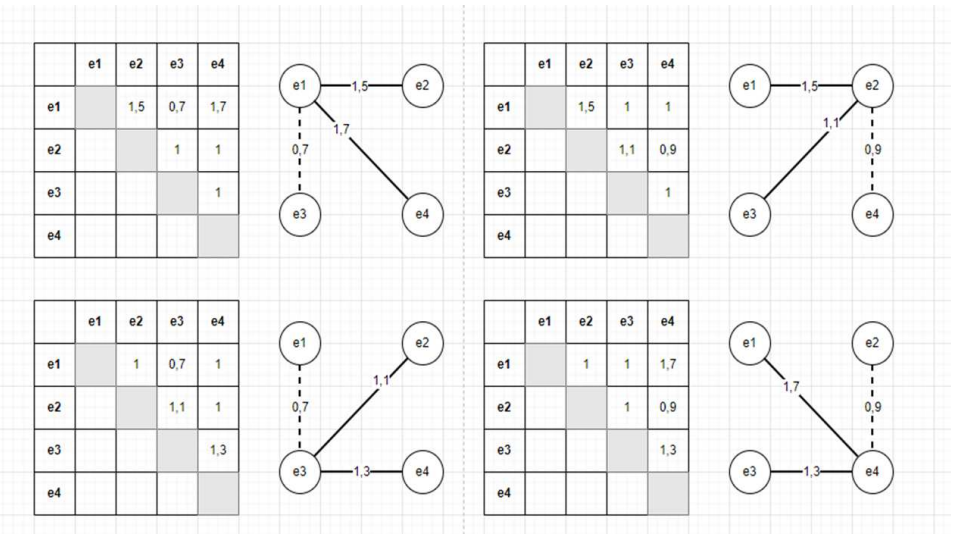
7. táblázat. Készségkövetelmények az egyes feladatokhoz

Az 1. ábra szerint, valamint a korábbi projekteredményeket és a tréner által készített DISC eredményeket figyelembe véve a következő szinergiamátrixot határoztuk meg (*8. táblázat*).

	e_1 (D)	e_2 (I)	e_3 (S)	e_4 (C)
e_1 (D)		1,5	0,7	1,7
e_2 (I)			1,1	0,9
e_3 (S)				1,3
e_4 (C)				

8. táblázat. A munkavállalók közötti szinergiamátrix.
(Csak felső háromszög van megadva)

Mivel a szinergiamátrix szimmetrikus és a főátlóban lévő értékek mindig 1-ek, elegendő a szinergiamátrix felső háromszögrészét megadni. A 8. táblázat azt mutatja, hogy pozitív szinergia van a domináns (D) és befolyásoló (I), a domináns (D) és a szabálytisztelő (C), a befolyásoló (I) és a kitartó (S), valamint a kitartó (S) és a szabálytisztelő (C) viselkedési stílusokkal rendelkező munkavállalók között, de negatív szinergiák vannak a domináns (D) és a kitartó (S), valamint a befolyásoló (I) és a szabálytisztelő (C) viselkedési stílusok között. Fontos, hogy a megadott szinergiamátrix összhangban van a szakirodalom javaslatával, amint azt a bevezetőben láttuk; azonban az általánosíthatóság érdekében a szimulációinkban minden pozitív és negatív szinergiaértékhez véletlenszerűen $\pm 20\%$ került hozzáadásra. Ha nem választunk vezetőt ehhez a csoporthoz, akkor a csoport szerepei önállóan kerülnek kiválasztásra. Ha azonban kis csoportban választunk ki vezetőt, akkor csak a csoportvezető és a többi munkavállaló szinergiája lehet domináns; ezért négy különböző, úgynevezett *domináns szinergiahálózatot* kapunk (lásd az *5. ábrát*, ahol csak a pozitív vagy negatív szinergiák láthatók).



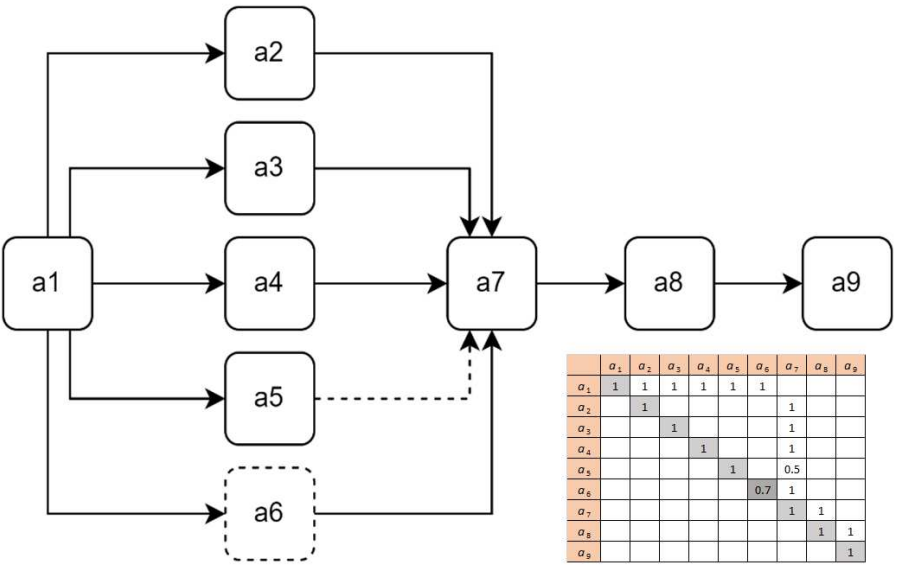
5. ábra. Lehetséges domináns szinergiahálózatok

A vizsgált vállalat az agilis projektmenedzsment megközelítések közül az ún. SCRUM[2] módszertant követi, amely szerint a projektet kisebb részprojektekre (sprintekre) kell felosztani. Az általunk vizsgált sprintnek kilenc feladata van: kialakítás megtervezése (a_1), A funkció implementálása és tesztelése (a_2), B funkció implementálása és tesztelése (a_3), C funkció implementálása és tesztelése (a_4), D funkció implementálása és tesztelése (a_5), az E extra funkció implementálása és tesztelése (a_6), a funkciók összehangolása és összehangolt tesztelése (a_7), hibajavítások (a_8) és a szoftver végleges integrációja (a_9). A vállalat szempontjából ez egy általános sprintnek felel meg, a feladatokat és az azok közti összefüggéseket pedig sablonszerűnek tekinthetjük. A feladatok prioritásait egy másik agilis módszertan, az ún. Dynamic System Development Method (DSDM) szerint határozzák meg, ahol a kötelező feladatok relatív prioritása, azaz a feladat pontértéke 1. Az alacsonyabb prioritásoknál alacsonyabb a feladatpontérték. A rugalmas függőségek a technológiából származtak. A logikai hálózatot a *6. ábra* adja meg.

A 6. ábrán látható, hogy csak egy kiegészítő feladat (a_6) és egy rugalmas függőség (a_5, a_7) található a projektben. A hozzárendelési részmátrix (**M** mátrix), amely a hozzárendelések maximális arányát tartalmazza, annak értéke minden cellában 1 volt, ami azt mutatta, hogy elvben minden mérnököt mindegyik feladathoz hozzá lehet rendelni, amelyet a keresztfunkcionalitás elve is előír.

A szimuláció négy plusz egy csoportkialakítást, négy célfüggvényt és három relatív korlátot ($C_x\% \in \{C_s\%, C_t\%, C_c\%\}$) vesz figyelembe.

[2]SCRUM nem egy rövidítés



	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
a_1	1	1	1	1	1	1			
a_2		1					1		
a_3			1				1		
a_4				1			1		
a_5					1		0.5		
a_6						0.7	1		
a_7							1	1	
a_8								1	1
a_9									1

6. ábra. Logikai struktúra és a precedenciamátrix

A relatív korlátokat a minimális és maximális követelmények alapján számítjuk ki az alábbiak szerint:

$$C_x\% = \frac{\mathrm{TPX}_{\max} - C_x}{\mathrm{TPX}_{\max} - \mathrm{TPX}_{\min}}, \tag{19}$$

ahol $C_x \in [\mathrm{TPX}_{\max}, \mathrm{TPX}_{\min}]$, TPX $\in$ {TPT,TPC,TPS}. Ha a munkavállalók nincsenek hozzárendelve egyik feladatra sem, $\mathrm{TPT}_{\max} = \infty$ vagy $\mathrm{TPC}_{\min} = 0$, akkor a minimális hozzárendelés a maximális hozzárendelés feleként lesz megadva ($e^{minw} = e^{maxw}/2$). Ily módon $\mathrm{TPT}_{\max}$ és $\mathrm{TPC}_{\min}$ számítható ki. Ebben a szimulációban $C_x\% \in \{0{,}5, 0{,}6, \ldots, 1{,}0\}$. 6^3-féle korlátozás lehetséges. Minden beállításnál 100 szimulációt adtunk meg, hogy figyelembe vegyük a szinergiaértékek becslésének érzékenységét. Ezért $6^3 \cdot 5 \cdot 4 \cdot 100 = 432\,000$ SMM-mátrixot kaptunk.

A szimulációk célja a (K) kutatási kérdés megválaszolása volt. Meghatároztuk az egyes szinergia-hálózatokat minden DISC szerint kijelölt vezetővel rendelkező csoport esetén, amelyeket összehasonlítottunk az autonóm kiválasztási módszerrel előállított csoporttal. Az esettanulmány alapján megfogalmazhatunk további konkrét részkérdéseket (RK).

RK$_1$ Van-e jelentősége a szinergiák figyelembevételének a projekt átfutási idejére és a költségeire vonatkozóan?

RK$_2$ A korlátozások és célfüggvények figyelembevételével melyik csoportszerep-kiválasztási módszerrel érhető el a legjobb megoldás?

RK$_3$ Melyik csoportszerep-kiválasztási módszer a legkevésbé érzékeny a korlátok változásaira?

5 Eredmények

A kutatási kérdés (K): *„A korlátok figyelembevételével, melyik csoportszerep-kiválasztási mechanizmus biztosítja a legrövidebb projektidőtartamot, illetve a legkisebb projektköltséget?"* megválaszolásához az első részkérdésre (RK_1)-re kell választ adnunk: *„Van-e jelentősége a szinergiák figyelembevételének a projekt átfutási idejére és a költségeire vonatkozóan?"*

A *9a. táblázat* a projektköltségre (TPC, 1000 EUR) és időtartamra (TPT, hét) vonatkozó leíró statisztikák eredményeit tartalmazza, a *9b. táblázat* pedig a kétmintás t-próba eredményeit mutatja be a szinergia figyelembevétele (syn) és figyelmen kívül hagyása (nosyn) esetén.

a. Leíró statisztika

	N	Átlag	Szórás	Std. hiba
TPT_{syn}	432 000	1,947	1,280	1,947
TPT_{nosyn}	432 000	2,025	1,331	2,025
TPC_{syn}	432 000	78,932	7,588	1,155
TPC_{nosyn}	432 000	83,078	8,079	1,232

b. Kétmintás t-próba

1. változó	2. változó	t-érték	Szab.fok	*p*
TPT_{syn}	TPT_{nosyn}	−1,000	431 999	0,317
TPC_{syn}	TPC_{nosyn}	−52,409	431 999	<0,001

9. táblázat. Projekt átfutási idők (TPT) és a projektköltségek (TPC) összehasonlítása

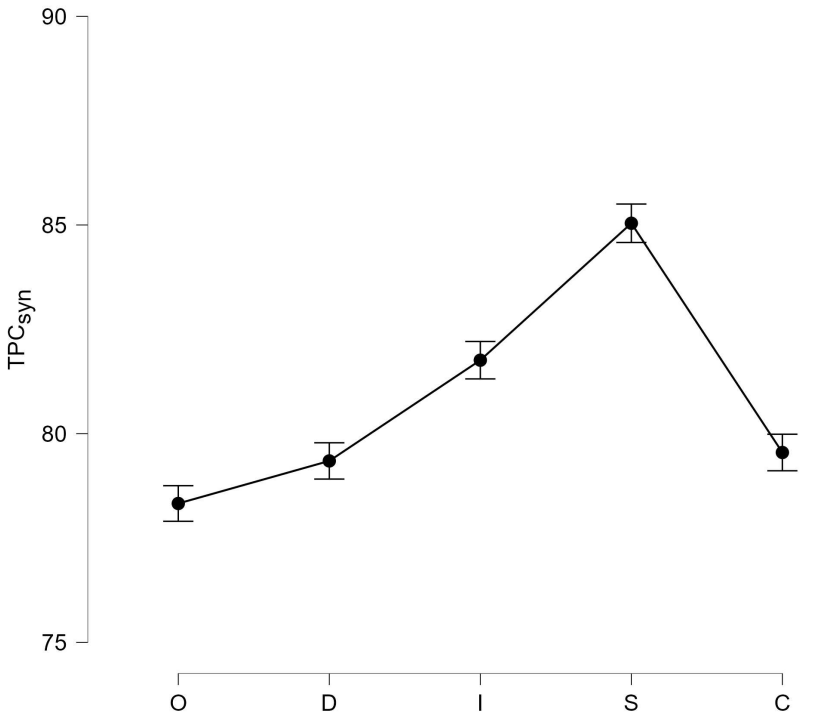
A 9a. táblázatban látható, hogy a szinergiák figyelembevétele csökkentheti a projekt időtartamát (TPT) és a projekt költségét (TPC). A TPT várható értéke több mint két hét a szinergiák figyelembevétele nélkül. A munkavállalók közötti szinergiát figyelembe véve a várható költségek 4146 euróval csökkenhetnek. A 9b. táblázat pedig azt mutatja, hogy csak a projektköltségek eltérése szignifikáns, ezért a továbbiakban csak a TPC-t vizsgáljuk.

Az RK_2 részkérdés *„A korlátozások és célfüggvények figyelembevételével melyik csoportszerep-kiválasztási módszerrel érhető el a legjobb megoldás?"* erősen kapcsolódik az eredeti kutatási kérdéshez (K). A *10. táblázat* a varianciaelemzés (ANOVA) eredményeit mutatja.

A 10. táblázat szerint elmondható, hogy csak a csoportszerep megválasztásának hatása szignifikáns a projekt összköltségének szempontjából. Ebben az esetben sem a célfüggvény, sem a korlátok változtatása nem szignifikáns.

ANOVA-TPC_{syn}	Négyzet-összeg	Szab. fok	Átlagos négyzetes eltérés	F	*p*
Csoportszerep	6 123 182	4	1 534 457	2,655	0,031
Célfüggvény választása	1 141 542	3	379 888	0,660	0,577
C_t%	3 254 545	5	649 867	1,129	0,342
C_c%	3 280 458	5	655 003	1,138	0,338
C_s%	2 430 254	5	485 072	0,842	0,519
Maradvány	$2,49 \times 10^{11}$	431 977	575 796		

10. táblázat. A csoportszerep, a különböző célfüggvények és korlátok megválasztásának hatása a projektköltségekre



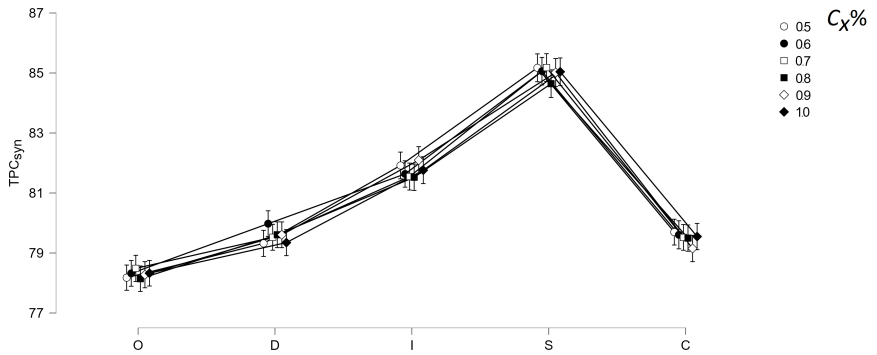
7. ábra. A projektköltségek alakulása a különféle csoportszerep-kiválasztási folyamatok esetén, ha $C_c\% = C_t\% = 1, C_s\% = 0$

A *7. ábra* a TPC_{syn} változását mutatja abban az esetben, ha a relatív költség- ($C_c\%$), idő- ($C_t\%$) és eredmény- ($C_s\%$) korlátokat figyelmen kívül hagyjuk. A legalacsonyabb projektösszköltség akkor jelentkezik, ha autonóm (O) csoportszerep-kiválasztási folyamatot alkalmazunk, míg a legnagyobb projektösszköltséget akkor kapjuk, ha a csoport vezetőjének viselkedési stílusa a DISC szerinti kitartó (S), aki ugyan kapcsolatorientált, a csapat ,,lelke (szülője)", de közben visszahúzódó, introvertált és rugalmatlan.

Bár a 10. táblázat azt mutatja, hogy a korlátoknak nincs szignifikáns hatása a projekt összköltségére, az RK_3 részkérdés szerint fontos megválaszolni, hogy: *,,Melyik csoportszerep kiválasztási módszer a legkevésbé érzékeny a korlátok változásaira?"* A Bartlett-teszt kimutatta, hogy a korlátok változásának nincs szignifikáns hatása a projekt összköltségének szórására.

A *8. ábra* azt is megmutatja, hogy a korlátok változása nem befolyásolja a projekt összköltségének konfidenciaintervallumát.

Az eredmények arra engednek következtetni, hogy ebben a projektben a minimális költségeket az autonóm csoportszerep esetén kaphatjuk, hiszen egyik csoportszerep-kiválasztási módszerrel megalkotott csoport sem érzékeny a relatív korlátok változására. Mivel valamennyi esetben legalább megengedett megoldást kapunk, így nincs szükség a projekt csoportstruktúrájának átalakítására sem.



8. ábra. Projektösszköltségek a különféle csoportszerep-kiválasztási folyamatok esetén, ha $C_c\% = C_t\% = 1 - C_s\%$

6 Összefoglalás

Ebben a tanulmányban az SSPSP probléma fejlesztését javasoltuk, amelynek segítségével modellezni lehet mind a puha, mind pedig a kemény készségeket. Megmutattuk, hogy a modellünk segítségével hogyan lehet megvizsgálni a csoportszerep-kiválasztási módszereket a projektidő és a projektköltség tekintetében. A probléma megoldására multikromoszómát alkalmazó HGA algoritmust javasoltunk. A szimulációk valós projekten alapultak, és az esettanulmány megmutatta, hogyan lehet kitölteni a javasolt módszer bemeneteként szolgáló módosított SMM mátrixot. A kutatási kérdésre (K) válaszolva az esettanulmányban rávilágítottunk arra, hogy az autonóm csoportkiválasztás biztosítja a legkisebb projektköltséget. Természetesen nem állítjuk, hogy ez az autonóm csoportszerep-kiválasztás minden esetben a legjobb megoldást adja, ugyanakkor olyan módszert mutattunk be, amelynek segítségével a különböző csoportszerep-kiválasztási módszerek összehasonlíthatók. A tanulmányban ismertetett esettanulmány bemutatja a javasolt módszer alkalmazhatóságát egy nagy, csúcstechnológiás autóipari vállalat K+F projektjének tervezésén keresztül. Az esettanulmány eredményei azt sugallják, hogy az autonóm csoportszerep-kiválasztási módszer jelentősen csökkentheti a projekt költségét, miközben megőrzi a projekt minőségét.

A tanulmány rámutat annak szükségességére, hogy az operációkutatási modellek esetén érdemes megfontolni a rugalmas projekttervezési módszerek sajátosságait. Ezen túlmenően a tanulmány új megközelítést kínál a puha és kemény készségek rugalmas projektekben történő modellezésére. Az operációkutatási modellekben pedig lehetőség nyílik a csoportszerep-kiválasztás projektteljesítményre gyakorolt hatásának mélyebb vizsgálatára. A menedzserek számára a javasolt módszer gyakorlati eszközt biztosít a projektek hatékonyabb menedzseléséhez. A módszer megmutathatja, hogy rugalmas projektek esetén az autonóm csoportszerep hogyan segíthet a költségek csökkentésében.

Ez a módszer más projektek tesztelésére is használható, ezért a későbbi kutatás során megválaszolhatjuk azt a kérdést, hogy mely projektek támaszkodjanak autonóm csoportkiválasztásra. Az autonóm csoportszerep-kiválasztási módszer használatával a vezetők optimalizálhatják a projekt időtartamát, költségeit, miközben megőrzik a projekt minőségét. Ez jelentős költségmegtakarításhoz és a projektmenedzsment hatékonyságának növeléséhez vezethet.

Köszönetnyilvánítás

A közlemény a TKP2021-NVA-10 számú projekt keretében a Kulturális és Innovációs Minisztérium Nemzeti Kutatási Fejlesztési és Innovációs Alapból nyújtott támogatásával, a 2021. évi Tématerületi Kiválóság Program pályázati program finanszírozásában valósult meg.

Irodalom

1. Al-Saqqa, S., Sawalha, S., AbdelNabi, H., (2020): Agile software development: Methodologies and trends. *International Journal of Interactive Mobile Technologies,* 14 (11), 246–270. DOI: https://doi.org/10.3991/ijim.v14i11.13269
2. Alba, E., Chicano, J. F., (2007): Software project management with GAs. *Information Sciences,* 177 (11), 2380–2401. DOI: https://doi.org/10.1016/j.ins.2006.12.020
3. Antoniou, A., (2019): Compatibility of small team personalities in computer-based tasks. *Challenges,* 10(1), DOI: https://doi.org/10.3390/challe10010029
4. Arashpour, M., Kamat, V., Bai, Y., Wakefield, R., Abbasi, B., (2018): Optimization modeling of multi-skilled resources in prefabrication: Theorizing cost analysis of process integration in off-site construction. *Automation in Construction,* 95, 1–9. DOI: https://doi.org/10.1016/j.autcon.2018.07.027
5. Aslam, W., Ijaz, F., (2018): A quantitative framework for task allocation in distributed agile software development. *IEEE Access,* 6, 15380–15390. DOI: https://doi.org/10.1109/ACCESS.2018.2803685
6. Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., és mtsai., (2001): The agile manifesto.
7. Belbin, R. M., (2012): *Team roles at work.* Routledge, Abingdon.
8. Belbin, R. M., Brown, V., (2022): *Team roles at work.* Routledge, London.
9. Birjandi, A., Mousavi, S. M., Hajirezaie, M., Vahdani, B., (2019): Optimizing and solving project scheduling problem for flexible networks with multiple routes in production environments. *Journal of Quality Engineering and Production Optimization,* 4 (1), 175–196. DOI: https://doi.org/10.22070/jqepo.2019.3870.1091
10. Browning, T. R., (2014): Managing complex project process models with a process architecture framework. *International Journal of Project Management,* 32 (2), 229–241. DOI: https://doi.org/10.1016/j.ijproman.2013.05.008
11. Capretz, L. F., (2003): Personality types in software engineering. *International Journal of Human-Computer Studies,* 58 (2), 207–214. DOI: https://doi.org/10.1016/S1071-5819(02)00137-4

12. Capretz, L. F., Ahmed, F., (2010): Why do we need personality diversity in software engineering? *ACM SIGSOFT Software Engineering Notes,* 35 (2), 1–11. DOI: https://doi.org/10.1145/1734103.1734111

13. Chang, C. K., Jiang, H.-y., Di, Y., Zhu, D., Ge, Y., (2008): Time-line based model for software project scheduling with genetic algorithms. Information and Software *Technology,* 50 (11), 1142–1154. DOI: https://doi.org/10.1016/j.infsof.2008.03.002

14. Chen, R., Liang, C., Gu, D., Leung, J. Y., (2017): A multi-objective model for multi-project scheduling and multi-skilled staff assignment for IT product development considering competency evolution. *International Journal of Production Research,* 55(21), 6207–6234. DOI: https://doi.org/10.1080/00207543.2017.1326641

15. Choi, K. S., Deek, F. P., Im, I., (2008): Exploring the underlying aspects of pair programming: The impact of personality. *Information and Software Technology,* 50 (11), 1114–1126. DOI: https://doi.org/10.1016/j.infsof.2007.11.002

16. Crawford, B., Soto, R., Johnson, F., Monfroy, E., Paredes, F., (nov 2014): A max-min ant system algorithm to solve the software project scheduling problem. *Expert Systems with Applications,* 41 (15), 6634–6645. DOI: https://doi.org/10.1016/j.eswa.2014.05.003

17. Curşeu, P. L., Meslec, N., Pluut, H., Lucas, G. J. M., (2015): Cognitive synergy in groups and group-to-individual transfer of decision-making competencies. *Frontiers in Psychology,* 6, 1375. DOI: https://doi.org/10.3389/fpsyg.2015.01375

18. Flores-Parra, J.-M., Castañón-Puga, M., Evans, R. D., Rosales-Cisneros, R., Gaxiola-Pacheco, C., (2018): Towards team formation using Belbin role types and a social networks analysis approach. In: *2018 IEEE Technology and Engineering Management Conference (TEMSCON).* IEEE, 1–6. DOI: https://doi.org/10.1109/TEMSCON.2018.8488386

19. Geissler, D. L., (2014): The black and white effect of being labeled: the influence of being labeled by the disc personality model from the perspective of the labeled participants. Master's thesis, University of Twente.

20. Hardiman, L. T., (1997): Personality types and software engineers. *Computer,* 30 (10), 10–10. DOI: https://doi.org/10.1109/MC.1997.625290

21. Kia, R., Shahnazari-Shahrezaei, P., Zabihi, S., (2016): Solving a multi-objective mathematical model for a multi-skilled project scheduling problem by cplex solver. In: *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM).* IEEE, 1220–1224. DOI: https://doi.org/10.1109/IEEM.2016.7798072

22. Kosztyán, Z. T., (2015): Exact algorithm for matrix-based project planning problems. *Expert Systems with Applications,* 42 (9), 4460–4473. DOI: https://doi.org/10.1016/j.eswa.2015.01.066

23. Kosztyán, Z. T., Bogdány, E., Szalkai, I., Kurbucz, M. T., (2022): Impacts of synergies on software project scheduling. *Annals of Operations Research,* 312, 883–908. DOI: https://doi.org/10.1007/s10479-021-04467-5

24. Kosztyán, Z. T., Harta, P., Szalkai, I., (2024): The effect of autonomous team role selection in flexible projects. *Computers & Industrial Engineering,* 190, 110079. DOI: https://doi.org/10.1016/j.cie.2024.110079

25. Kosztyán, Z. T., Novák, G., Jakab, R., Szalkai, I., Hegedűs, C., (2023): A matrix-based flexible project-planning library and indicators. *Expert Systems with Applications,* 216, DOI: https://doi.org/10.1016/j.eswa.2022.119472

26. Kosztyán, Z. T., Szalkai, I., Kurbucz, M. T., (2021): Szinergiák hatása a szoftverfejlesztési projektekben. *Alkalmazott Matematikai Lapok,* 38 (1), 105–126. DOI: https://doi.org/10.37070/AML.2021.38.1.08
27. Larson, Jr., J. R., (2007): Deep diversity and strong synergy: Modeling the impact of variability in members' problem-solving strategies on group problem-solving performance. *Small Group Research,* 38 (3), 413–436. DOI: https://doi.org/10.1177/1046496407301972
28. Larson, Jr., J. R., (2009): *In search of synergy in small group performance.* Psychology Press, New York. DOI: https://doi.org/10.4324/9780203848784
29. LePine, J. A., Buckman, B. R., Crawford, E. R., Methot, J. R., (2011): A review of research on personality in teams: Accounting for pathways spanning levels of theory and analysis. *Human Resource Management Review,* 21 (4), 311–330. DOI: https://doi.org/10.1016/j.hrmr.2010.10.004
30. Looi, Q. E., See, S. L., Tay, C. S., Ng, G. K., (2011): Understanding users by their DISC personality through interactive gaming. In: *HCI International 2011–Posters' Extended Abstracts: International Conference,* HCI International 2011, Orlando, FL, USA, July 9-14, 2011, Proceedings, Part I 14. Springer, 312–316. DOI: https://doi.org/10.1007/978-3-642-22098-2_63
31. Luna, F., González-Álvarez, D. L., Chicano, F., Vega-Rodríguez, M. A., (2014): The software project scheduling problem: A scalability analysis of multi-objective metaheuristics. *Applied Soft Computing,* 15, 136–148. DOI: https://doi.org/10.1016/j.asoc.2013.10.015
32. Lykourentzou, I., Antoniou, A., Naudet, Y., Dow, S. P., (2016): Personality matters: Balancing for personality types leads to better outcomes for crowd teams. In: *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing.* 260–273. DOI: https://doi.org/10.1145/2818048.2819979
33. Maghsoudlou, H., Afshar-Nadjafi, B., Niaki, S. T. A., (2017): Multi-skilled project scheduling with level-dependent rework risk; three multi-objective mechanisms based on cuckoo search. *Applied Soft Computing,* 54, 46–61. DOI: https://doi.org/10.1016/j.asoc.2017.01.024
34. Marston, W. M., (1928): Review of a short outline of comparative psychology. *Journal of Abnormal and Social Psychology,* 23 (2), 256–257. DOI: https://doi.org/10.1037/h0068416
35. Meslec, N., Curşeu, P. L., (2015): Are balanced groups better? Belbin roles in collaborative learning groups. *Learning and Individual Differences,* 39, 81–88. DOI: https://doi.org/10.1016/j.lindif.2015.03.020
36. Mostert, N. M., (2015): Belbin – the way forward for innovation teams. *Journal of Creativity and Business Innovation,* 1, 35–48.
37. Myers, I. B., (1962): *The Myers-Briggs Type Indicator: Manual (1962).* Consulting Psychologists Press. DOI: https://doi.org/10.1037/14404-000
38. Myszkowski, P. B., Laszczyk, M., Nikulin, I., Skowroński, M., (2019): Imopse: a library for bicriteria optimization in multi-skill resource-constrained project scheduling problem. *Soft Computing,* 23(10), 3397–3410. DOI: https://doi.org/10.1007/s00500-017-2997-5
39. Nikolaou, I., Oostrom, J. K., (2015): *Employee recruitment, selection, and assessment.* Contemporary Issues for Theory and Practice (1st ed.). Psychology Press. DOI: https://doi.org/10.4324/9781315742175
40. Nosek, J. T., (1998): The case for collaborative programming. *Communications of the ACM,* 41, 105–108. DOI: https://doi.org/10.1145/272287.272333

41. Orense, R. C., Ocampo, R., (2015): Correlation on the DISC personality profile and leadership styles of the student leaders of CAS in A.Y, 2014-2015. *The Bedan Journal of Psychology,* 1, 90–101.

42. Oza, N., Fagerholm, F., Münch, J., (2013): How does kanban impact communication and collaboration in software engineering teams? In: *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE).* IEEE, 125–128. DOI: https://doi.org/10.1109/CHASE.2013.6614747

43. Paetsch, F., Eberlein, A., Maurer, F., (2003): Requirements engineering and agile software development. In: *WET ICE 2003. Proceedings.* Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. IEEE, 308–313. DOI: https://doi.org/10.1109/ENABL.2003.1231428

44. Peslak, A. R., (2006): The impact of personality on information technology team projects. In: *Proceedings of the 2006 ACM SIGMIS CPR conference on computer personnel research.* Forty four years of computer personnel research: achievements, challenges & the future. 273–279. DOI: https://doi.org/10.1145/1125170.1125233

45. Pieterse, V., Leeu, M., van Eekelen, M., (2018): How personality diversity influences team performance in student software engineering teams. In: *2018 Conference on Information Communications Technology and Society (ICTAS).* IEEE, 1–6. DOI: https://doi.org/10.1109/ICTAS.2018.8368749

46. Reeves, C. R., Wright, C. C.: Genetic algorithms and the design of experiments. In: Davis, L. D., De Jong, K. A., Vose, M. D., Whitley, L. D. (eds.) *Evolutionary Algorithms.* IMA Volumes in Mathematics and its Applications, vol. 111, Springer, New York, 207–226. DOI: 10.1007/978-1-4612-1542-4_12

47. Scullard, M., Baum, D., (2015): *Everything DiSC Manual.* John Wiley & Sons.

48. Sfetsos, P., Stamelos, I., Angelis, L., Deligiannis, I., (2009): An experimental investigation of personality types impact on pair effectiveness in pair programming. *Empirical Software Engineering,* 14, 187–226. DOI: https://doi.org/10.1007/s10664-008-9093-5

49. Shen, X.-N., Minku, L. L., Marturi, N., Guo, Y.-N., Han, Y., (2018): A q-learning-based memetic algorithm for multi-objective dynamic software project scheduling. *Information Sciences,* 428, 1–29. DOI: https://doi.org/10.1016/j.ins.2017.10.041

50. Stylianou, C., Andreou, A. S., (2012): A multi-objective genetic algorithm for software development team staffing based on personality types. In: *Artificial Intelligence Applications and Innovations.* 8th IFIP WG 12.5 International Conference, AIAI 2012, Halkidiki, Greece, Proceedings, Part I 8. Springer, 37–47. DOI: https://doi.org/10.1007/978-3-642-33409-2_5

51. Vega-Velázquez, M. Á., García-Nájera, A., Cervantes, H., (2018): A survey on the software project scheduling problem. *International Journal of Production Economics,* 202, 145–161. DOI: https://doi.org/10.1016/j.ijpe.2018.04.020

52. Walle, T., Hannay, J. E., (2009): Personality and the nature of collaboration in pair programming. In: *2009 3rd International Symposium on Empirical Software Engineering and Measurement.* IEEE, 203–213. DOI: https://doi.org/10.1109/ESEM.2009.5315996

53. Wang, M., Liu, G., Lin, X., (2022): Dynamic optimization of the multi-skilled resource-constrained project scheduling problem with uncertainty in resource availability. *Mathematics,* 10. DOI: https://doi.org/10.3390/math10173070

54. Whitehead, J., (2007): Collaboration in software engineering: A roadmap. In: *Future of Software Engineering (FOSE'07).* IEEE, 214–225. DOI: https://doi.org/10.1109/FOSE.2007.4

55. Winter, B., (2015): *Agile performance improvement: The new synergy of agile and human performance technology.* Apress, Berkeley. DOI: https://doi.org/10.1007/978-1-4842-0892-2

56. Wongwai, N., Malaikrisanachalee, S., (2011): Augmented heuristic algorithm for multi-skilled resource scheduling. *Automation in Construction,* 20(4), 429–445. DOI: https://doi.org/10.1016/j.autcon.2010.11.012

57. Wysocki, R. K., (2019): *Effective project management: traditional, agile, extreme.* John Wiley & Sons. DOI: https://doi.org/10.1002/9781119562757

58. Xiao, J., Ao, X.-T., Tang, Y., (2013): Solving software project scheduling problems with ant colony optimization. *Computers & Operations Research,* 40(1), 33–46. DOI: https://doi.org/10.1016/j.cor.2012.05.007

AUTONOMOUS GROUP ROLE SELECTION IN FLEXIBLE PROJECTS

This paper proposes a modified synergy-based software project scheduling problem (SSPSP) to model group roles and synergies between workforces in flexible projects. The proposed model can manage flexible projects, synergies between workforces, and the difference between hard and soft skills. The study also explores the effect of autonomous group role selection, which is crucial in agile projects. The proposed method is evaluated in a real-life case study and compared with existing, non-synergy-based project scheduling methods.

The core of the paper is the proposed modified SSPSP, which can handle various skills and performance levels, as well as a synergy structure to model synergies between different behavioral types. The matrix-based model presented in this study can analyze the effect of autonomous group role selection, which is crucial in agile projects. The proposed method is evaluated in a real-life case study, which demonstrates the effectiveness of the proposed method in the R&D sector of a large high-tech automotive company. The results show that the autonomous group role selection mechanism can significantly reduce project duration while maintaining project quality. The main finding of this study is that positive synergy between workforces can reduce the makespan of the project, and the best solution is given in autonomous group role selection. The case study presented in this manuscript demonstrates the effectiveness of the proposed method in reducing project duration while maintaining project quality. The proposed method provides a practical tool for project management, which can lead to significant cost savings and increased efficiency in project management. The novelties of this study are the proposed modified SSPSP, which can handle various skills and performance levels, as well as a synergy structure to model synergies between different behavioral types. The matrix-based model presented in this study can analyze the effect of autonomous group role selection, which is crucial in agile projects. The proposed method is evaluated in a real-life case study, which demonstrates the effectiveness of the proposed method in reducing project duration while maintaining project quality. The proposed method provides a practical tool for project management, which can lead to significant cost savings and increased efficiency in project management.

In conclusion, this study proposes a modified SSPSP to model group roles and synergies between workforces in flexible projects. The proposed method is evaluated in a real-life case study and compared with existing, non-synergy-based project scheduling methods. The results show that the autonomous group role selection

mechanism can significantly reduce project duration while maintaining project quality. The proposed method provides a practical tool for project management, which can lead to significant cost savings and increased efficiency in project management.

Keywords: software project scheduling; synergy-based modeling; DISC behavioral types; autonomous group role selection. *JEL codes:* C88, D83, M15, O32