Multidiszciplináris tudományok, 9. kötet. (2019) 4 sz. pp. 301-304 **https://doi.org/10.35925/j.multi.2019.4.29**

ERŐFORRÁS-KORLÁTOS KÖRNYEZETBEN PÁRHUZAMOSAN FUTÓ PROJEKTEK TÖBBCÉLÚ ÜTEMEZÉSE

Mihály Krisztián
tanársegéd, Miskolci Egyetem Gépészmérnöki és Informatikai Kar
Alkalmazott Informatikai Intézeti Tanszék
3515 Miskolc, Miskolc-Egyetemváros, e-mail: altmihaly@uni-miskolc.hu

Kulcsár Gyula
egyetemi docens, Miskolci Egyetem Gépészmérnöki és Informatikai Kar
Alkalmazott Informatikai Intézeti Tanszék
3515 Miskolc, Miskolc-Egyetemváros, e-mail: iitkgy@uni-miskolc.hu

Absztrakt

Napjainkban a projekt alapú feladatvégrehajtás egyre több iparágban jellemző megközelítésnek számít, a termékéletciklusát tekintve a termékfejlesztéstől, a gyártáson át a terméktámogatásig. Valós környezetben több, egymástól céljaiban eltérő, párhuzamosan végrehajtandó projekt osztozik ugyanazon erőforrásokon. A feladatok végrehajtási sorrendjének kulcs szerepe van abban, hogy összességét tekintve a különböző célmutatókat minél jobban elérjük. Munkánk során olyan modelleket és projekt-ütemező eljárásokat vizsgáltunk, amelyek az ilyen típusú feladatok végrehajtására alkalmasak.

Kulcsszavak: *projekt-ütemezés, generálási séma, többcélúság, párhuzamos projekt*

Abstract

Nowadays the project-based task execution is widely used in many industrial areas, in different project lifecycle phases, from the design to the manufacturing and support. In real industrial environments typically several parallelly executed projects are sharing the same resources. The order of tasks execution is a key factor to achieve the defined goals. Our paper presents new scheduling models and methods to solve the presented problems.

Keywords: *project scheduling, generation scheme, multi-objective, multi-project*

1. Bevezetés

A projektek ütemezése jól kutatott területnek számít, mely jelentős elméleti háttérrel, eredményekkel és gyakorlatban alkalmazott, bizonyítottan hatékony eljárásokkal rendelkezik. A projektet olyan, előre ismert elvégzendő feladatok listájaként tekintjük, mely feladatok végrehajtása előre ismert célok megvalósítására szolgál. A projekt célja, hogy a feladatok végrehajtására rendelkezésre álló erőforrások használatával, az előre definiált korlátok figyelembevételével úgy oldjuk meg a feladatokat, hogy az előre definiált célokat minél jobban teljesítsük. [1]

A projektütemezés feladata jól ismert, de jellegét tekintve – a speciális esetektől eltekintve – az NP nehéz feladatok közé soroljuk, mivel az alkalmazott modellek és megoldásukra kifejlesztett algoritmusok használhatósága nagyban függ a valós életben tapasztalt jellemzők figyelembevételétől. [2]

2. Erőforrás-korlátos környezet

Több ismert, ütemező modell és algoritmus tekinti az erőforrásokat korlátlannak, s ily módon bizonyítottan optimális ütemezési megoldást tudnak nyújtani bizonyos célfüggvényekre. Ilyen többek közt a kritikus út módszere (*CPM – Critical Path Method*).

Erőforrás-korlátos környezetben az ismert feladatok végrehajtására nem áll rendelkezésre annyi erőforrás, hogy a felmerülő kapacitásszükségletet bármely időpillanatban biztosan ki tudjunk szolgálni. Az erőforrás-korlátos projektütemezési feladatot (*RCPSP – Resource Constraint Project Scheduling*) az alábbi formalizmussal tudjuk definiálni [3] [4] [5]:

- Adott az elvégzendő feladatok egy előre ismert listája: $T = \{1,2,3, \dots ,n\}$
- Adottak előre ismert erőforrás típusok: $K = \{1,2,3, \dots ,m\}$
  - Egy erőforrás típusnak korlátozott kapacitása áll rendelkezésre, melyet bármely időpontban meg tudunk határozni: R_k, $k \in K$
  - Az erőforrásokat megújulónak tekintjük, egy feladat végrehajtása után a feladat végrehajtásához használt kapacitás újra rendelkezésre áll.
- A feladatoknak 2 típusú korlátot határozunk meg:
  - Minden feladatnak ismert a feladat végrehajtásának elkezdéséhez szükséges más feladatok halmaza: P_j
    A feladat csak akkor ütemezhető, ha minden i feladat ($i \in P_j$) végrehajtásra került.
  - Egy feladat végrehajtáshoz szükség lehet egy, vagy több erőforrástípusból, erőforrásonként meghatározott kapacitásra. A feladat akkor hajtható végre, ha az igényelt kapacitás szükséglet erőforrástípusonként egyszerre kielégíthető.

Egy ütemezést végrehajthatónak nevezünk, ha a feladatokat a megadott sorrendben végrehajtva az erőforráskorlátok és megelőzési relációk teljesülnek.

3. Párhuzamosan futó projektek

Az RCPSP probléma egyik kiterjesztéseként modelleztük azt az esetet, amikor az erőforrásokat nem egy, hanem több, párhuzamosan futó projekt végrehajtásához is hozzá kell rendelnünk.

Az összerendelés egyik lehetséges megközelítése, hogy bevezetünk egy virtuális, globális kezdő ($GS \in T$) és befejező feladatot ($GT \in T$). A GS feladatot minden projekt előfeltétel nélküli feladatának megelőző feladatként vezetjük be, a GT feladatnak minden olyan feladat előfeltétele, amely az eredeti definíció szerint nem megelőző feladata más feladatnak. Ezzel a párhuzamosan futó projektek feladata egy egy-projektes feladattá redukálható, de az ütemezés jóságának meghatározásához egy globális, projektek összeségét egyszerre optimáló célfüggvény(ek) meghatározása szükséges. Ez a gyakorlatban sokszor nehézkes.

A projektek virtuális összekötése helyett modellünkben a párhuzamosan futó projektek saját célfüggvényekkel rendelkeznek, melyeket egymástól függetlenül lehet meghatározni. A projektek egy közös feladatlistából kerülnek meghatározásra, mely során a feladatok megelőzési relációja projektenként meghatározott.

4. Többcélúság

Egy projektütemezés jóságát célfüggvények alkalmazásával tudjuk leírni. Két végrehajtható projekt ütemezés közül azt tekintjük jobbnak egy célfüggvény szempontjából, amely függvényt alkalmazva a

kapott függvény értéke kisebb (illetve célfüggvénytől függően nagyobb). Az ütemezés célja, hogy a lehető legjobb ütemezést megtaláljuk, azaz a lehetséges végrehajtható ütemezések közül azokat, amelyekre a célfüggvény értéke minimális (illetve célfüggvénytől függően maximális).

Általánosságban tekintve egy projekt jóságának kifejezése általában nehezen fogalmazható meg egyetlen célfüggvénnyel. Több célfüggvény alkalmazásának egyik lehetséges módja, hogy egy közös mutatót alakítunk ki, melyben az egyes célfüggvényeket különböző súlyokkal látjuk el és a végső célfüggvény az egyedi célfüggvények súlyozott összegeként jelenik meg.

5. Alkalmazott modell és ütemező

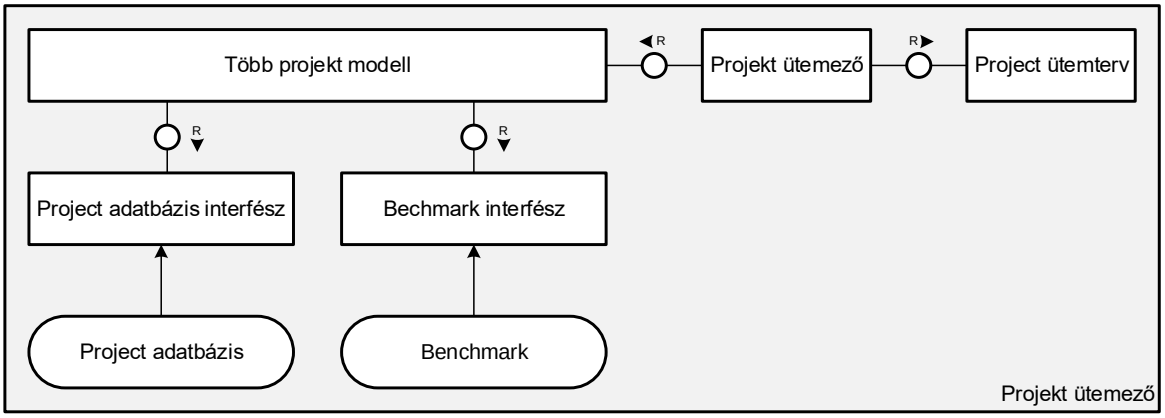
A kialakított modellben több projektet tudunk megadni, melyek közös feladatokon és erőforrásokon osztoznak. A feladatok korlátait az első megvalósításban projektektől függetlenül határozzuk meg. Célfüggvények definiálására projektenként van mód, mely során minden célfüggvény saját súllyal rendelkezhet. A lehetséges célfüggvények készlete előre meghatározott.

Az ütemező algoritmus a generálási sémákon alapul. A generálási sémák elve, hogy iteratívan hoz létre egy végrehajtható ütemezést. Egy üres ütemtervből indul ki és minden iterációs lépésben egy újabb feladatot ad hozzá az ütemtervhez a meghatározott korlátok figyelembevételével. Az iteráció első lépése, hogy meghatározza a feladat és a már létrehozott ütemterv alapján azon feladatok listáját, amelyek az ütemezés következő lépésében végrehajthatóak.

Működési elvét tekintve megkülönböztetünk soros és párhuzamos generálási sémát. A különbség alapja a végrehajtható feladatok meghatározásában van. Munkánk első implementációjában soros generálási sémát alkalmaztunk, amikor egy feladat végrehajthatónak tekinthető, ha minden indításához szükséges feladat már végrehajtásra került. A felépítő algoritmus a döntési halmazból választ egy feladatot, majd azt az ütemezéshez adja. Az ütemező alapötlete az, hogy a döntési halmazból választás során, heurisztikus jelleggel figyelembe veszi a projekt(ek)re és a projekt feladataira definiált célfüggvényeket.

6. Összegzés

A megoldásunk egy különálló modulként kerül kialakításra.



1. ábra. *Projekt ütemező moduláris felépítése*

A modul fő elemei:
- Több projekt modell
  - A párhuzamosan futó projektek reprezentálása, projektenként definiált célfüggvényekkel és célfüggvény paraméterekkel
  - A feladatok reprezentálása, a feladatok korlátaival és egyedi, feladat szintű célfüggvényeivel és célfüggvény paramétereivel
  - Az erőforráson reprezentálása, az erőforrások korlátainak leírása
- Projekt ütemező: Feladata a definiált projekt ütemezése a paraméterként meghatározott ütemezési eljárás alapján. Az első implementációban az ütemezési eljárás a soros generálási sémát követi.
- Projekt ütemterv: az egyes feladatok indítási idejét határozza meg

A modell implementálását SAP Netweaver 7.20 környezetben végeztük el, de a megoldás nem tartalmaz SAP specifikus implementációt. Az implementáció tesztelésére két megközelítést alkalmaztunk:

1. Összevetés ismert, optimumot biztosító algoritmussal: Ismeretes, hogy 2 gépes ütemezési problémára a Johnson algoritmus optimális megoldást ad [6]. A keresési eljáráson alapuló modellünk kis projektek esetén nagy valószínűséggel (>95%) megtalálta az optimumot.
2. RCPSP benchmark: RCPSP problémákra és azokra adott megoldásokra rendelkezésre állnak elérhető benchmark feladatok. [7-9] A benchmark feladatokra futtatva az ütemezőt összehasonlítottuk az ismert legjobb eredményekkel. Azt tapasztaltuk, hogy kis feladatok esetén döntő többségben (>70%) az ismert optimum megtalálásra került.

Irodalom

[1] Amol, S.: Resource Constrained Multi-Project Scheduling with Priority Rules & Analytic Hierarchy Process, Proc. Eng. 2014, 69:725-734. **https://doi.org/10.1016/j.proeng.2014.03.048**

[2] Garey M R, Johnson D S.: Computers and intractability a guide to the theory of NP completeness 1979, W. H. Freeman and Company

[3] Kolisch R, Hartmann S.: Heuristic Algorithms for the Resource-Constrained Project Scheduling Problem: Classification and Computational Analysis. Proj. Scheduling. Intern. Series in Op. Res. & Mgmg. Scn. 1999, 14:147-178. **https://doi.org/10.1007/978-1-4615-5533-9_7**

[4] Mohanthy R P, Siddiq M K.: Multiple projects multiple resources-constrained scheduling: Some studies. Int. Jrnl. of Prod. Res. 1989, 27:261-280. **https://doi.org/10.1080/00207548908942546**

[5] Deckro R F, Winkofsky E P, Hebert J E, Gagnon R.: A decomposition approach to multi-project scheduling. Europ. Jrnl. of Oper. Res. 1991, 51:110-118. **https://doi.org/10.1016/0377-2217(91)90150-T**

[6] Johnson, Donald B.: Efficient algorithms for shortest paths in sparse networks, Journal of the ACM 1977, 24(1):1–13. **https://doi.org/10.1145/321992.321993**

[7] R. Kolisch, A. Sprecher. PSPLIB - A project scheduling library. European Journal of Operational Research, 1996, 205-216. **https://doi.org/10.1016/S0377-2217(96)00170-1**

[8] Project Scheduling Problem Library, 2019, http://www.om-db.wi.tum.de/psplib/getdata.cgi?mode=sm

[9] Schutt A., Feydy T., Stuckey P.J. (2013) Explaining Time-Table-Edge-Finding Propagation for the Cumulative Resource Constraint. In: Gomes C., Sellmann M. (eds) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. CPAIOR 2013. Lecture Notes in Computer Science, vol 7874. Springer, Berlin, Heidelberg. **https://doi.org/10.1007/978-3-642-38171-3_16**