AZ NIIF SZUPERSZÁMÍTÓGÉP FELHASZNÁLÁSÁNAK STATISZTIKAI ELEMZÉSE ADAPTÍV GRID ÜTEMEZŐ MEGVALÓSÍTÁSÁHOZ

Lipovits Ágnes, lipovitsa@szt.vein.hu
Kovács Előd, kovacse@almos.vein.hu
Veszprémi Egyetem, Matematikai és Számítástechnikai Tanszék
Juhász Zoltán, juhasz@irt.vein.hu
Veszprémi Egyetem, Információs Rendszerek Tanszék

1. Bevezetés

Az elosztott rendszerek napjainkban egyre nagyobb szerepet kapnak. Fontos feladattá vált az Internethez hasonló, nagyméretű, széleskörű elérést biztosító olyan elosztott rendszer létrehozása, amely földrajzi távolság, hardver- és szoftver-független, heterogén, és a felhasználók számára biztosítja a koherens, egységesnek látszó végrehajtó környezetet. Egy ilyen rendszer fejlesztésének célja különféle szolgáltatások biztosítása, nagy számítási igényű és adat-intenzív feladatok párhuzamos megvalósítása, elosztott erőforrásokat használó problémák megoldása.

A földrajzilag elosztott, nagyméretű rendszereknek a létrehozása, programozása és felhasználása sok, eddig megoldatlan problémát vet fel, melyek megoldása a működéshez alapvetően fontos. Egyik megoldásra váró feladat heterogén, nagy rendszerekben is alkalmazható grid ütemezési algoritmusok és módszerek kifejlesztése.

Az optimális ütemezéshez az erőforrások hardver paramétereinek és a várakozási sor hosszának ismeretén kívül elengedhetetlenül fontos, hogy becsülni tudjuk az adott várakozási sorokhoz tartozó teljes végrehajtási időt (várakozási idő + futási idő). Mivel ez időben változó eloszlású valószínűségi változó, a megvalósítás csak adaptív rendszer segítségével kivitelezhető. Az adaptív grid ütemező modul egy rendszerre jellemző végrehajtási idő, mint valószínűségi változó eloszlását becsüli, figyelembe véve a várakozási sor hosszát, mint paramétert. A kapott valószínűség eloszlások alapján a várakozási sorokat (grid erőforrás) a várható befejezési idő bizonyos kvantilisei szerint rangsorolja. Az ütemező ennek alapján a várakozási sor hossza helyett a modul által előállított értéket veszi figyelembe a feladat futtatási helyének meghatározásához.

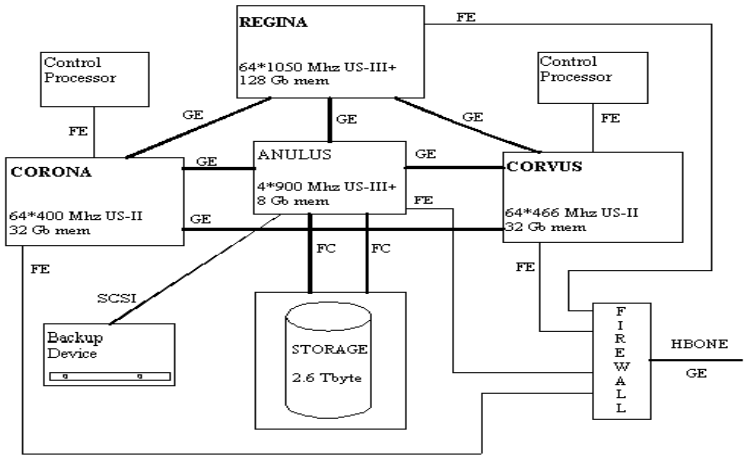
A statisztikai alapokon nyugvó, intelligens ütemezési modell (ISSM) létrehozásának első lépése a rendszer szignifikáns bemenő paramétereinek meghatározása. Célszerű ehhez a már működő szuperszámítógépek, klaszterek működésének karakterisztikáját elemezni. Ebben a cikkben az NIIF szuperszámítógép 2003. április és 2005. szeptember közötti naplózott adatait elemezzük, és az eredményeket összehasonlítjuk más rendszerek működési paramétereivel. [2, 5, 7, 8] A vizsgálat kiterjed az erőforrások használatára (memória, CPU), a párhuzamosság mértékére, a feladatok beérkezésének időbeli periodicitására és különösen hangsúlyosan a várakozási és futási időre. Bemutatjuk az elemzés eredményeit, illetve az egyes paraméterekből levonható következtetéseket. Végezetül ismertetjük a tervezett ütemező rendszer architektúra tervét és a modell részletes működését.

2. Az NIIF szuperszámítógép

Magyarország legnagyobb teljesítményű számítógépe az NIIF Iroda által üzemeltetett 148 processzoros szuperszámítógép. Feladata, hogy kiszolgálja a hazai tudományos kutatás és felsőoktatás igényeit. Biztosítja a különlegesen nagy számítási teljesítményt igénylő tudományos kutatások informatikai hátterét, lehetővé teszi a HPC (High Performance Computing) alkalmazások hazai futtatását. A szuperszámítógép szolgáltatás 2001-ben indult, naplózott adatok a 2003. április és 2005. szeptember közötti időszakról állnak rendelkezésünkre. Összesen 157638 db job-ról készült mentés.

A Szuperszámítógép Központ legfőbb erőforrása a két SUN 15K szuperszámítógép, melyek közös SGE (Sun Grid Engine [19]) klaszterbe vannak szervezve. A szuperszámítógépeket egy szintén SUN

gyártmányú 4 processzoros E480 szerver, az anulus.niif.hu szolgálja ki, amely többek között fájl szolgáltatást, adatmentési szolgáltatást, HPC master szerepet biztosít a többi gép számára. A master host egyik legfontosabb feladata a felhasználó jobjainak megfelelő szétosztása. A két, egyenként 72 processzoros szuperszámítógép – a codex.niif.hu és a regina.niif.hu – feladata a feladatok végrehajtása. A két SUN 15K gépet kiegészíti még az Eötvös Loránd Tudományegyetemen működő, 64 processzoros E10K, a corvus, és a Gödöllői Agrártudományi Egyetemen működő, az előbbihez hasonló felépítésű corona is [10].



1. ábra Az NIIFI Szuperszámítógép Központ architektúrája [10]

3. A naplózott adatok elemzése

Az SGE qacct parancsa segítségével lekérdezhetőek a következő – futás során naplózott – adatok: *qname*, *hostname*, *group*, *owner*, *project*, *department*, *jobname*, *jobnumber*, *taskid*, *account*, *priority*, *qsub_time*, *start_time*, *end_time*, *granted_pe*, *slots*, *failed*, *exit_status*, *ru_wallclock*, *ru_utime*, *ru_stime*, *ru_maxrss*, *ru_ixrss*, *ru_ismrss*, *ru_idrss*, *ru_isrss*, *ru_minflt*, *ru_majflt*, *ru_nswap*, *ru_inblock*, *ru_oublock*, *ru_msgsnd*, *ru_msgrcv*, *ru_nsignals*, *ru_nvcsw*, *ru_nivcsw*, *cpu*, *mem*, *io*, *iow*, *maxvmem*. A továbbiakban a rendszer működése és az ütemezés szempontjából lényeges jellemzőket elemezzük.

I. failed: sikertelenül végrehajtott feladatok

A naplózott adatok szerint a jobok 96,3%-a sikeresen lefutott. Az összes adat 21,9%-a rossz elküldési idővel (Thu Jan 1 01:00:00 1970) volt naplózva, ezért az adatbázisból töröltük. (Ezek mind hpc feladatok, és többszörösen tároltak.) A maradék 118 652 db job közül néhány (főleg a naplózási időszak elején) korábbi futási mint küldési idővel szerepelt. Ezek törlése után 118 506 adatsor maradt. Az **exit_status** mező szerint a hibátlanul lefutott jobok száma: 110 406.

II. qname: a várakozási sorok kihasználtsága

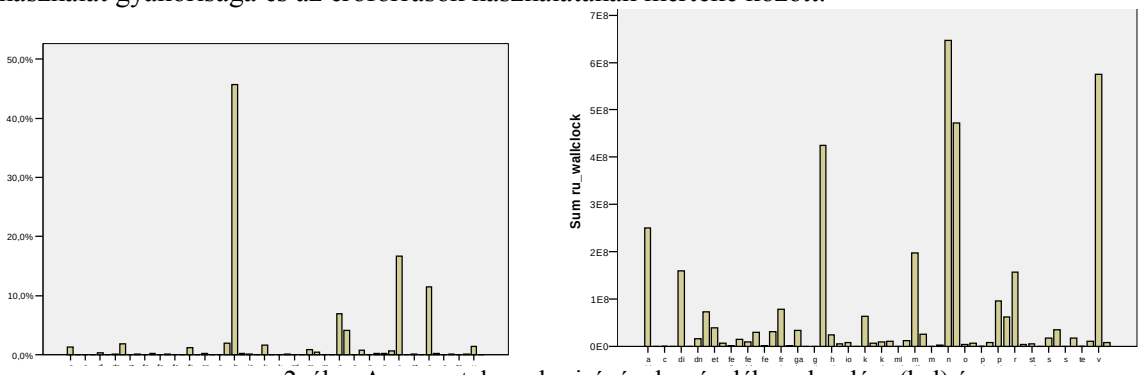
Az 1. táblázat a rendelkezésre álló queue-kat és használatuk gyakoriságát mutatja. Jól látható, hogy mind a négy gépen létezik egy .in végű interaktív sor, melynek feladata a feladatok futtathatóságának ellenőrzése, illetve magas prioritású, de maximum 24 órán át futó feladatok fogadása, illetve a .q végű normál queue. A kumulatív gyakoriság azt is mutatja, hogy 98,7%-ban a normál sorokat használják.

1. táblázat A végrehajtási sorok használata

	gyakoriság	százalék	Kumulatív százalék
codex.q	40723	34,4	34,4
corona.q	16223	13,7	48,1
corvus.q	20533	17,3	65,4
regina.q	39509	33,3	98,7
codex.in	84	,1	98,8
corona.in	1251	1,1	99,9
corvus.in	33	,0	99,9
regina.in	150	,1	100,0
Total	118506	100,0	

III. group, **owner**: a job tulajdonosa és a hozzátartozó csoport.

A gyakorisági diagramról leolvasható, hogy az egyes csoportok nagyon különböző mértékben használják az infrastruktúrát. A futási időt ábrázoló diagram mutatja, hogy nincs összefüggés a használat gyakorisága és az erőforrások használatának mértéke között.



2. ábra A csoportok gyakoriságának százalékos eloszlása (bal) és a csoportok összes futásidejének eloszlása (jobb).

IV. priority: a job prioritás.

A job prioritás értéke -1024 és 1023 között változhat, az alapprioritás minden job esetén 0. Ez az alapbeállítás queue-tól függően változtatható. A regina.in queue alapprioritása például magasabb: 10. A futtatandó program prioritását a felhasználó csak csökkentheti (növelni kizárólag az SGE adminisztrátor privilégiuma). A kimutatásokból látható, hogy az alapbeállítást nem változtatták a használat során. Ennek oka, hogy bár az SGE többféle ütemezési stratégiát is ismer, a jelenlegi beállítás felhasználói szintű ütemezést tesz lehetővé. Ez annyit jelent, hogy minden felhasználó egyenlő feltételekkel, egyenlő prioritással helyezhet el feladatokat. Nem lehetséges tehát az, hogy míg egy felhasználó sok feladattal terheli a klasztert, addig a többiek várakozásra kényszerülnek.

V. granted_pe: párhuzamos futtatási környezet.

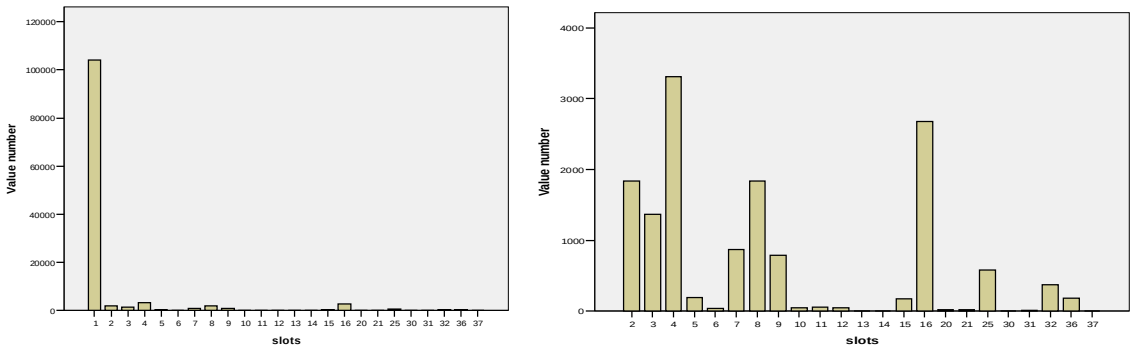
A kimutatásból látszik, hogy a futtatott programoknak több mint 83,3%-a nem használ párhuzamos környezetet. A kereszttábla mutatja, hogy a maradék 16,7%-ból egy processzoron fut az MPI támogatást használó programok 23,8%-a, a fork() rendszerhívást használó programok 41,2%-a.

2. táblázat A párhuzamos és a soros programok aránya a különböző futtatási környezetekben

	fork		hpc		none	pvm
	Count	%	Count	%	Count	Count
serial	1638	41,2%	3758	23,8%	98671	
parallel	2337	58,8%	12019	76,2%		83
Total	3975	100,0%	15777	100,0%	98671	83

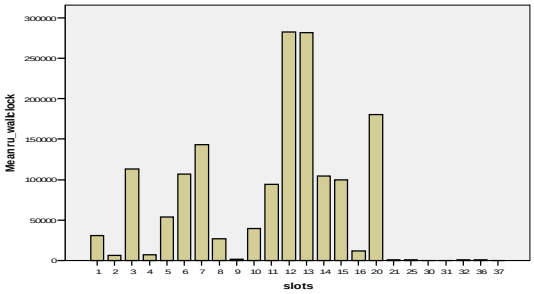
VI. slots: a párhuzamosság mértéke

Az összes job 87,8%-a csupán egy processzort használ, így nem él a szuperszámítógépek párhuzamos futtatási lehetőségével. A 3.b ábra mutatja, hogy a maradék 22,7% milyen eloszlást mutat. Az NIIF szuperszámítógép job méret karakterisztikája nem hasonlít a korábban elemzettekhez, ahol dominánsak a kétprocesszoros alkalmazások [5, 7, 8].

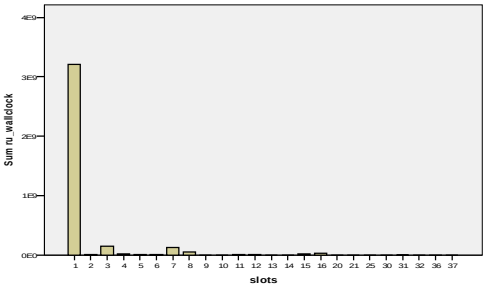


3. ábraA jobok eloszlása processzorszám szerint (bal) illetve
csak a párhuzamos jobok processzorszám eloszlása (jobb)

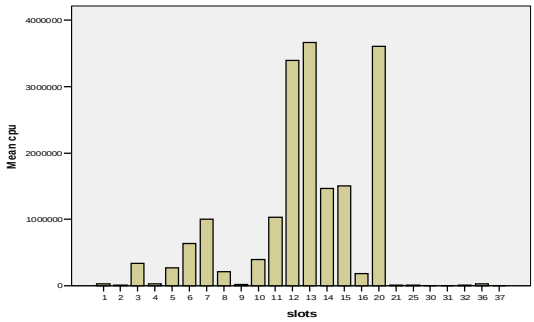
A 4. ábra négy diagramja az átlagos és összes futási idő, valamint az átlagos és összes processzor használat (futási idő * a használt processzorok száma) eloszlását mutatja a processzorszám függvényében.



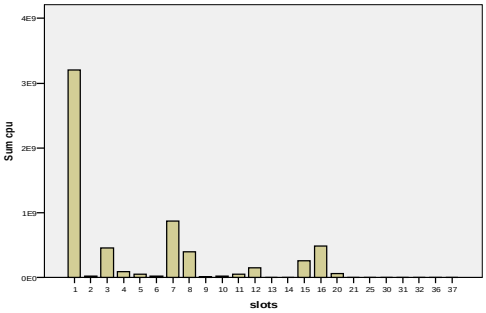
a) átlagos futási idő per processzor szám



b) összes futási idő per processzor szám



c) átlagos processzor használati idő
per processzor szám

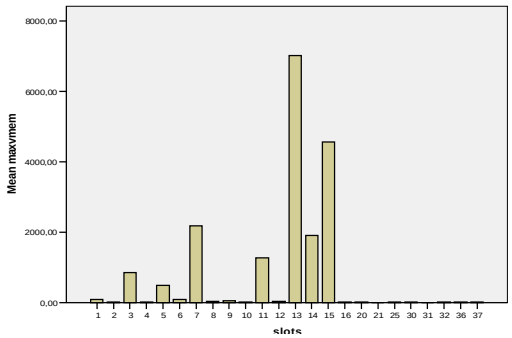


d) összes processzorhasználati idő
per processzor szám

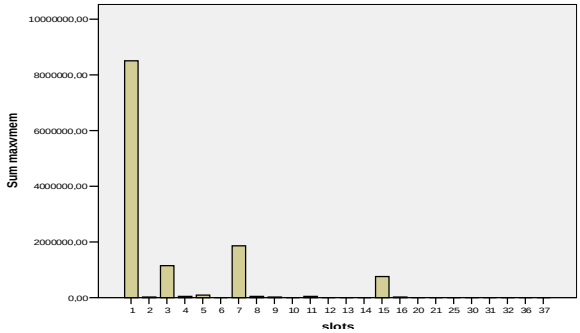
4. ábra

A következő ábrák segítségével megvizsgáljuk, hogy az átlagos, az összes virtuális memória és az átlagos memóriahasználat hogyan oszlik el a processzorszám függvényében. A szekvenciális programok nagy arányából adódik, hogy az erőforrások használatának összesítésében is domináns a súlyuk. Az átlagos memória- és processzorhasználatnál kiemelkedő értéket a 13 processzoros alkalmazásoknál találunk, illetve magas értéket adnak még a processzorhasználat esetében a 12 és a 20 processzoros feladatok. Látható, hogy a legtöbb egyprocesszoros job esetében, egyik erőforrás használata sem indokolja, hogy szuperszámítógépen futtassák.

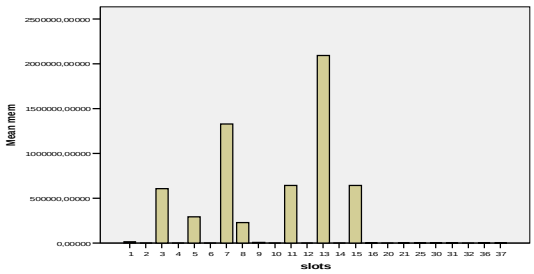
Korrelációs vizsgálatot végeztünk a következő paraméterek között: processzor használat, átlagos memória használat, párhuzamosság mértéke és a maximális virtuális memória. Mivel ezek a változók nem normális eloszlásúak, a Spearman-féle rangkorrelációs együtthatókat számoltuk. Az eredmény (3. táblázat) hasonló más rendszereknél kapott értékekhez, csak a memória és a párhuzamosság mértéke közti együtthatónál kaptunk lényegesen különböző (negatív) értéket.[1, 8]



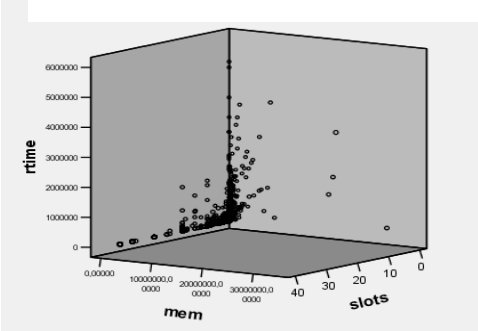
a) átlagos maximális virtuális memória használat per processzorszám



b) összes maximális virtuális memória használat per processzorszám



c) átlagos memóriahasználat per processzorszám



d) A memóriahasználat, a futási idő és a processzorszám kapcsolata

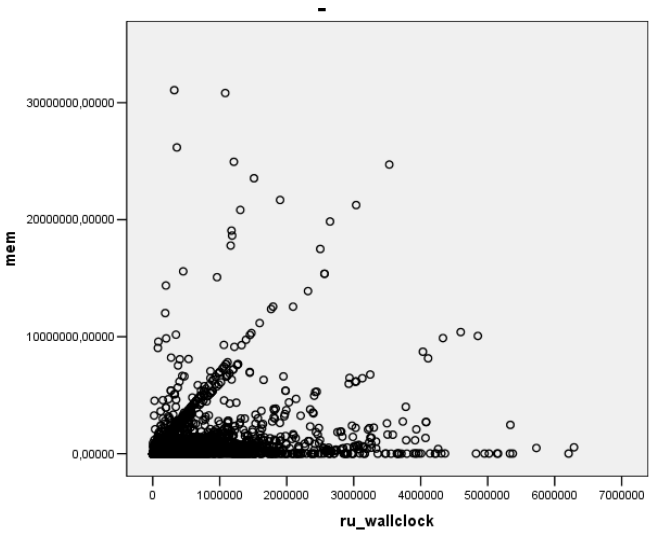
5. ábra

3. táblázat Korrelációs együtthatók értéke a használt erőforrások között

Correlations

			cpu	mem	slots	maxvmem
Spearman's rho	cpu	Correlation Coefficient	1,000	,589**	,122**	,103**
		Sig. (2-tailed)	.	,000	,000	,000
		N	11848	11848	11848	11799
	mem	Correlation Coefficient	,589**	1,000	-,360**	,570**
		Sig. (2-tailed)	,000	.	,000	,000
		N	11848	11848	11848	11799
	slots	Correlation Coefficient	,122**	-,360**	1,000	-,101**
		Sig. (2-tailed)	,000	,000	.	,000
		N	11848	11848	11848	11799
	maxvmem	Correlation Coefficient	,103**	,570**	-,101**	1,000
		Sig. (2-tailed)	,000	,000	,000	.
		N	11799	11799	11799	11799

**. Correlation is significant at the 0.01 level (2-tailed).



6. ábra A futási idő és az átlagos memóriahasználat kapcsolata

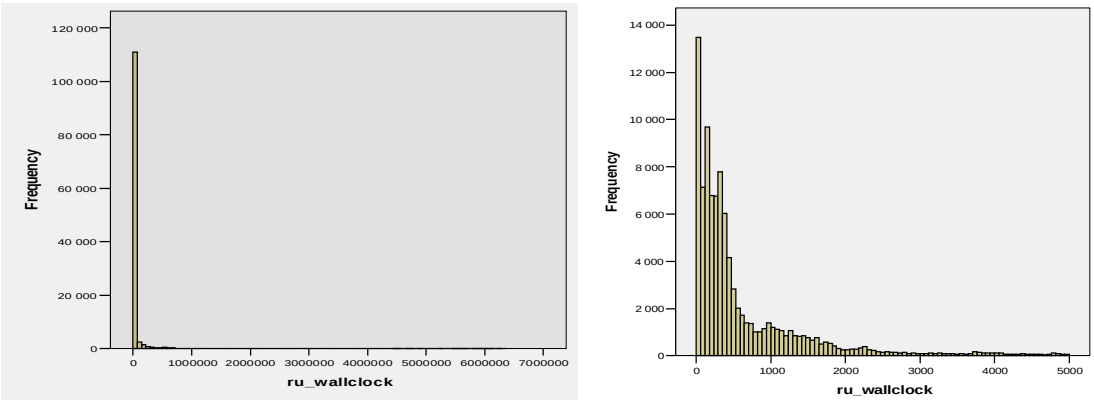
VII. qsub_time, **start_time**, **end_time**: a sorba helyezés ideje, a futás kezdete és vége.

Ezekből a jellemzőkből számolható a várakozási idő (qtime), a futási idő (rtime) és az elküldéstől a befejezésig eltelt idő (alltime). Az eredményeket a 4. táblázat tartalmazza.

4. táblázat Az összes futási idő, a várakozási idő és a futási idő statisztikai jellemzői

		alltime	qtime	rtime
N	Valid	118506	118506	118506
Mean		98342,83	67494,00	30848,84
Median		5596,00	3405,00	431,00
Mode		15	13	1
Std. Deviation		296071,765	220889,886	171763,479
Variance		87658490059,507	48792341921,003	29502692824,0
Range		7202393	7178652	6287621

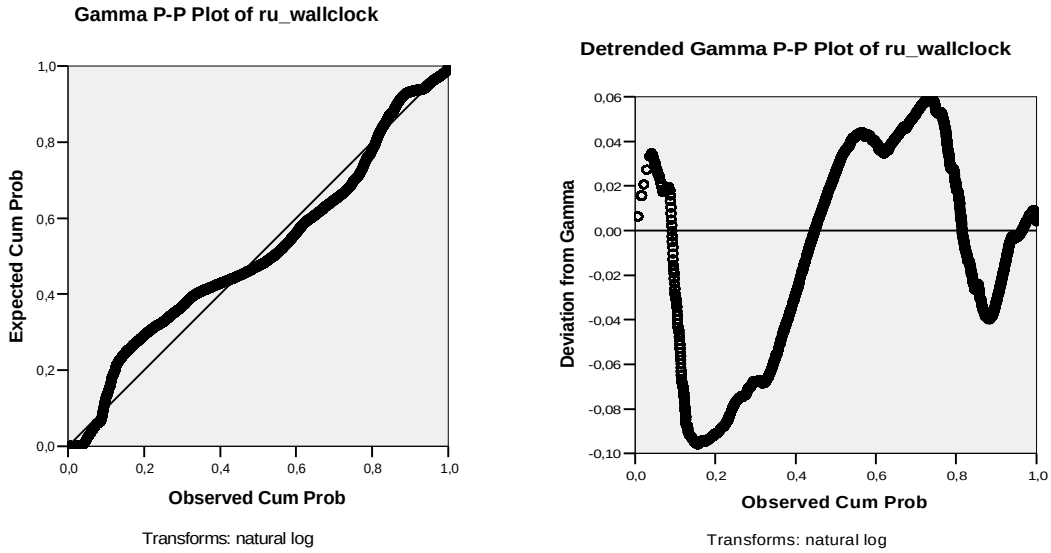
A futási időre illesztett eloszlások közül logaritmikus transzformáció után legjobban a Gamma eloszlás illeszkedett. ($\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx$, $f(x) = \frac{\lambda^\alpha x^{\alpha-1}}{\Gamma(\alpha)} e^{-\lambda x}$, ahol $\alpha, \lambda > 0$) A kapott paraméterek: $\alpha = 6{,}130, \lambda = 0{,}932$



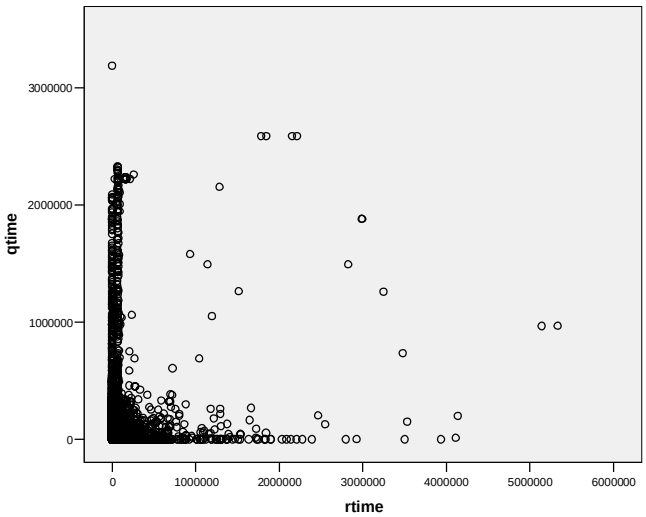
7. ábra A futási idők eloszlása a teljes tartományon (balra) illetve
a futási idők eloszlása a nagy adatok (10%) figyelembe vétele nélkül (jobbra)

A szuperszámítógépek működésének fontos jellemzője az úgynevezett expansion factor (E_f), ami a teljes végrehajtási idő (várakozási idő + futási idő) és a futási idő hányadosa. $E_f > 1$ minden esetben, és minél kisebb, annál hatékonyabb a rendszer működése. Az NIIF szuperszámítógép esetén ez az érték

átlagosan 605,8, a hozzá tartozó szórás D=28394,4. Ha figyelembe vesszük a párhuzamos processzorhasználatot, akkor 289,6-re csökken az átlag és D=18761,5.



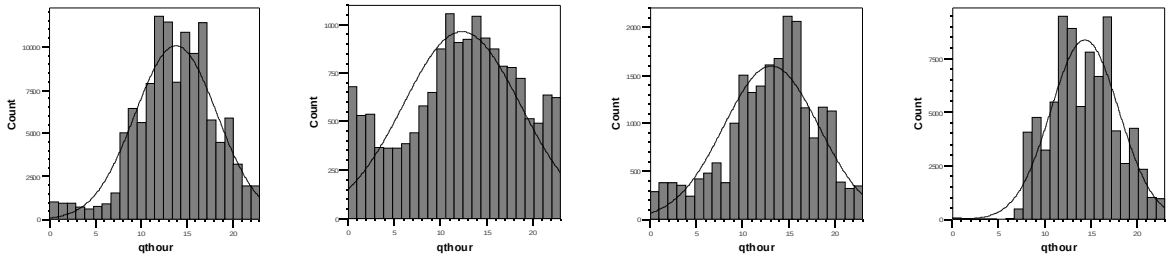
8. ábra A futási idő logaritmikus transzformációjára illesztett Gamma eloszlás (balra) illetve az illeszkedés legnagyobb eltérése (jobbra)

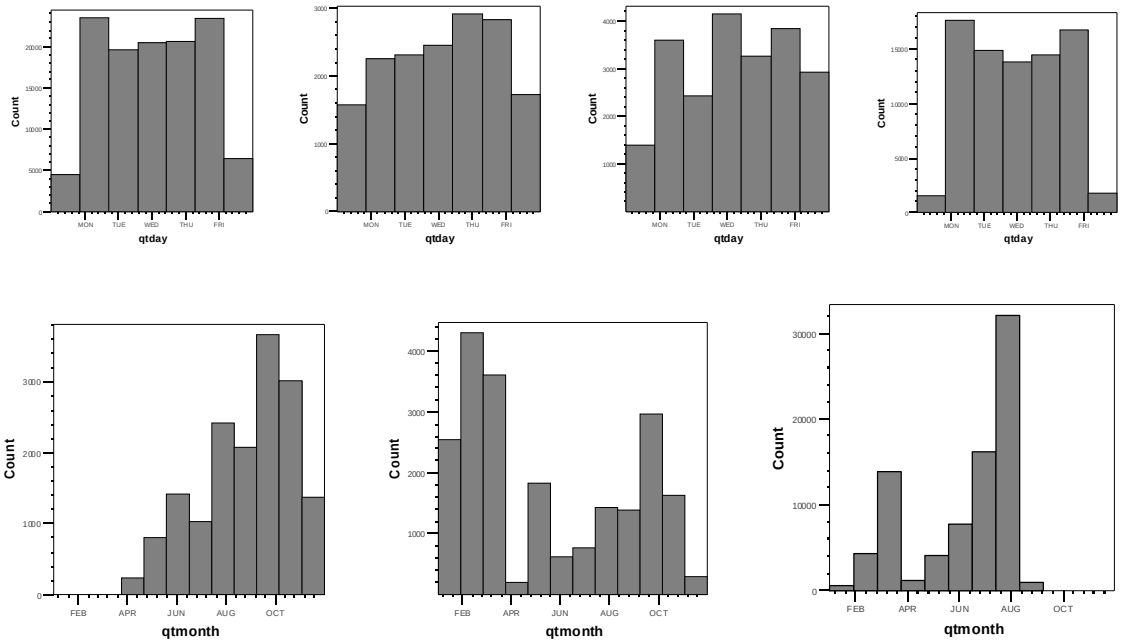


9. ábra A várakozási idő és a futási idő közti összefüggés

VIII. A feladatok beérkezésének időbeli periodicitása

A napi periodicitás minden rendszernél megfigyelhető. Definiálni lehet a csúcsidőszakot, és a különböző időszakokban a beérkezett jobok számára eloszlást lehet illeszteni. Az első ábra az összes adat eloszlását mutatja, majd az évek szerinti felbontást. (2003, 2004, 2005) A napi ciklusnál a várt munkaidő szerinti megoszláson kívül különböző ingadozások is megfigyelhetőek. Az utolsó sorozat mutatja a havi eloszlást évek szerint. Érdekesség, hogy áprilisban minden évben nagyon alacsony a használat.

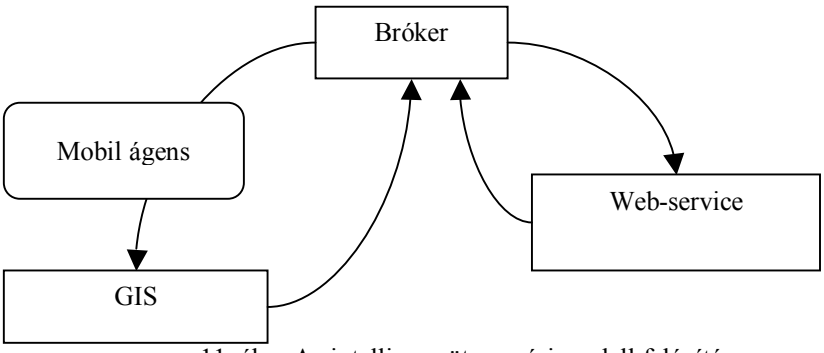




10. ábra A jobok beérkezésének napi eloszlása (balról jobbra: összes, 2003, 2004, 2005) (felső) illetve heti eloszlás (balról jobbra: összes, 2003, 2004, 2005) (középső) illetve havi eloszlás (balról jobbra: 2003, 2004, 2005) (alsó)

4. Az intelligens ütemezési modell bemutatása

A tapasztalat azt mutatja, hogy a kapott eloszlások típusa és paraméterei időben változnak, ezért a várakozási idő becsléséhez egy olyan mobil adaptív ágens kifejlesztése szükséges, amelyik az újonnan kapott adatok alapján képes a megfelelő típusú eloszlást és a hozzá tartozó paramétereket meghatározni [14, 15, 16, 17]. Az első lépésben figyelembe vett eloszlások típusa: exponenciális, hiperexponenciális, lognormális, reciprok, gamma, Pareto, illetve Weibull.[5, 7, 18] Mivel a feldolgozáshoz nagyméretű adatfájl szükséges, a hatékonyság érdekében célszerű ezt a részfeladatot kódáthelyezéssel megoldani. Az algoritmus jellegéből adódóan elég a gyenge mobilitás biztosítása.[11] Az implementációt célszerű úgy megvalósítani, hogy az összes platform támogassa. A logfájl helyi feldolgozása után csak az eredményül kapott paramétereket kell a bróker felé továbbítani, a kommunikációs csatorna minimális igénybevételével. A bróker az újonnan kapott adatokkal meghívhatja az adott szolgáltatást (grid service), mely előállítja az új eloszlásfüggvényt. Az új függvény előállításához szükséges a várakozási sor hossza, az eloszlás típusa és paramétere(i). A teljes futási idő, mint valószínűségi változó független, azonos eloszlású és paraméterű valószínűségi változók összege lesz. ($\eta_k = \xi_1 + \xi_2 + \mathbf{K} + \xi_k$, k a várakozási sor hossza) η_k eloszlásának meghatározása bizonyos esetekben – pl. ξ_i-k exponenciális eloszlásúak – nem nehéz feladat. Bonyolultabb esetekben viszont csak az eloszlás bizonyos jellemzőit határozzuk meg. A várhatóérték például: $E(\eta_k) = k \cdot E(\xi_1)$. A kvantilisek becslése pedig szimulációval történhet.



11. ábra Az intelligens ütemezési modell felépítése

5. Összefoglalás

Az utóbbi években több működő rendszer (szuperszámítógép, klaszter) futási adatait összegyűjtötték, elemezték, és a működésüket modellezték. A kapott eredmények segítenek az új architektúra modellek tervezésénél, és az ütemezési stratégiák kialakításánál. Az NIIF szuperszámítógép karakterisztikája néhány vonatkozásban eltér a más rendszereknél kapott jellemző viselkedéstől. Ilyen a párhuzamosság mértéke. A tapasztalat szerint leggyakoribb a 2 processzor használata. A jobok futásának néhány jellemző adatára eloszlásokat illesztettek, de ezek különböző rendszereknél eltérőek lehetnek. Mi az ütemezési modellünkhöz szükséges futási időre vizsgáltuk meg az eloszlások illeszkedését. Először a más rendszereknél működő eloszlásokkal próbálkoztunk: lognormális, Pareto, exponenciális és Weibull. De eredményül azt kaptuk, hogy a gamma eloszlás adja a legjobb illeszkedést logaritmusos transzformáció után.

A kapott eredmények azt mutatják, hogy növelné az ütemezés hatékonyságát, ha megvalósulna a modellünkben lévő modul, amely képes egy rendszerre jellemző futási idők eloszlásának és az eloszlás paramétereinek meghatározására. Jövőbeli célunk, hogy a naplózott adatok segítségével a jobok várakozási sorának hosszát kiszámoljuk, és segítségével a modellünk szerint számolt és a tényleges futási idők értékeit összehasonlítsuk.

Irodalom:

[1] D. G. Feitelson Memory usage in the LANL CM-5 Workload. In D. G. Feitelson and L. Rudolph, editors, Job Scheduling Strategies for Parallel Processing, Springer-Verlag, 1997.

[2] W. Smith, I. Foster, and V. Taylor. Predicting application run times using historical information. Lecture Notes in Computer Science, 1459:122–136, 1998.

[3] W. Smith, V. Taylor, and I. Foster. Using run-time predictions to estimate queue wait times and improve scheduler performance. Lecture Notes in Computer Science, 1659:202–219, 1999.

[4] W. Smith and P. Wong. Resource selection using execution and queue wait time predictions. Technical Report NAS-02-003, NASA Ames Research Center, 2002.

[5] S.-H. Chiang and M. K. Vernon. Characteristics of a large shared memory production workload. Lecture Notes in Computer Science, 2221: 159–187, 2001.

[6] D. G. Feitelson. Workload modeling for performance evaluation. Lecture Notes in computer Science, 2459:114–141, 2002.

[7] U. Lublin and D. G. Feitelson. The workload on parallel supercomputers: modeling the characteristics of rigid jobs. J. Parallel and Distributed Comput., 63(11): 1105–1122, 2003.

[8] Hui Li, David Groep, Jeff Templon, and Lex Wolters. Predicting job start times on clusters. In proceedings of 4th IEEE/ACM International Symposium on Cluster Computing and the Grid, IEEE Press, 2004.

[9] PDM - Toolkit for mining performance data in the Grid. http://www.liacs.nl/home/hli/pdm/ .

[10] http://www.niif.hu

[11] Andrew S. Tanenbaum – Maarten van Steen: Elosztott rendszerek, alapelvek és paradigmák, Panem 2004

[12] Jarek Nabrzyski, Jennifer M. Schopf, Jan Weglarz: Grid resource management, State of the Art and Future Trends, Kluwer Academic Publishers, Boston/Dordrecht/London, 2003

[13] Allan Snavely, Greg Chun, Henri Casanova University of California, San Diego; Rob F. Van der Wijngaart, Michael A., Frumkin, NASA Ames Research Center: Benchmarks for Grid Computing: A Review of Ongoing Efforts and Future Directions, ACM SIGMETRICS Performance Evaluation Review, 2003

[14] Multi-agent Computing System in a Heterogeneous Network, Uhruski P., Grochowski M., Schaefer R., PARELEC 2002 Conference

[15] Grochowski M., Schaefer R., Uhruski P.:Diffusion Based Scheduling in the Agent-Oriented Computing System, Lecture Notes in Computer Science, 3019 (2004), 97-104.

[16] László Csaba Lőrincz, Tamás Kozsik, Attila Ulbert, Zoltán Horváth: A method for job scheduling in Grid based on job execution status, Multiagent and Grid Systems, 2005

[17] S.S. Manvia, M.N. Birjeb and Bhanu Prasadc: An Agent-based Resource Allocation Model for computational grids, 2005 IEEE International Conference on Services Computing Vol-1 pp. 311-314

[18] Workload Characteristics of the DAS-2 Supercomputer, Hui Li, Lex Wolters, David Groep, 2004

[19] http://www.sun.com/software/grid/