

**Párhuzamos és metaszámítási alkalmazások
makrólépésekre alapozott
korszerű hibakeresési és -javítási módszerei**

Tézisfüzet

Készítette: Lovas Róbert

Témavezető: Dr. Kacsuk Péter (MTA SZTAKI)

*Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar*

Budapest, 2005

I. Bevezetés

A nem-determinisztikus párhuzamos programok helyességével kapcsolatos hibakeresés és –javítás (debugging) igen időigényes és fáradtságos feladat, különösen interaktív módon. Ebben az esetben, a szoftvermérnököknek szembe kell néznie a próba-hatással, a megismételhetőség és a teljesség problémájával, valamint a nagy állapottérrel, amit be kell járni a szoftverfejlesztési ciklus hibakeresési és –javítási fázisában. Továbbá az új, nagy számítási teljesítményt igénylő (HPC) alkalmazások megkövetelik a heterogén és földrajzilag elosztott erőforrások kihasználását is, ami újabb kihívásokat támaszt az alkalmazásfejlesztő eszközökkel szemben.

Amíg a hibakeresés és –javítás (valamint a tesztelés) fontosságát elismerik a párhuzamos szoftverfejlesztés területén, még mindig nem találhatunk széles körben elterjedt és felhasználóbarát hibakeresési és –javítási módszereket, eszközöket ezen a területen. A bemutatott munka megkísérel túllépni a meglévő hibakeresési és –javítási megoldások korlátain, és ötvözi azok hagyományos módszereit a párhuzamos és metaszámítási programok automatikus modellezésével és formális ellenőrzésével.

A bemutatott munka szorosan kötődik két szoftverfejlesztő keretrendszerhez: az MTA SZTAKI által kidolgozott P-GRADE párhuzamos programozási környezethez, valamint az Emory University, az Oak Ridge National Laboratory és a University of Tennessee által fejlesztett HARNESS metaszámítási rendszerhez.

A P-GRADE [4] integrált környezet grafikus megoldást nyújt a párhuzamos alkalmazások fejlesztéséhez és végrehajtásához klasztereken, szuperszámítógépeken és Grid rendszereken. A P-GRADE jelentősen felgyorsítja a meglévő szekvenciális programok újratervezését, magában foglalva a hierarchikus tervezési fázist egy hibrid grafikus nyelv, a GRAPNEL segítségével, a hibakeresést és –javítást, a tesztelést, az on-line monitorozást, a teljesítményanalízist, valamint a vizualizációs fázisokat. Párhuzamos Virtuális Gép (PVM) használata esetén a P-GRADE végrehajtó környezete a hosszan futó GRAPNEL alkalmazásokhoz dinamikus terheléskegyenlítést biztosít a teljesen automatikus ellenőrzőpont (checkpoint) készítő és migrációs mechanizmusai segítségével.

A HARNESS [K] egy metaszámítási rendszer, amely kísérletet tett, hogy túllépjen a hagyományos elosztott számítási keretrendszerek (mint pl. a PVM [AA]) korlátozott rugalmasságán azzal, hogy egy egyszerű, de mégis hatékony architektúrális modellt definiál az ún. szoftver-hátlap (*software backplane*) koncepcióra alapozva. A HARNESS metaszámítási keretrendszer alapvető absztrakciója az Elosztott Virtuális Gép (DVM), ahol nemcsak az erőforrások száma, hanem maguk a DVM által felkínált szolgáltatások is újrakonfigurálhatóak.

II. Kapcsolódó kutatások

A szakirodalom alapján [A], az elosztott hibakeresési és –javítási metodikák osztályokba sorolhatóak az általuk nyújtott felhasználói támogatás alapján: milyen szinten segítik a felhasználót a globális predikátumok specifikációja és detektálása területén, valamint az elosztott programozási hibák felderítésében.

A “*Távoli szekvenciális folyamatok interaktív hibakeresése*” módszer a tradicionális szekvenciális hibakeresési parancsok kiterjesztésén alapul. Ez az alapvető megközelítés lehetővé teszi az egyes távoli szekvenciális folyamatok online

megfigyelését és végrehajtásának vezérlését. Ezt a funkciót majdnem minden létező kereskedelmi és akadémiai elosztott hibakereső és –javító eszköz támogatja.

A reprodukálhatóság problémáját a „Nyomkövetés, visszajátszás és hibakeresés” módszer úgy oldja meg, hogy az első végrehajtás során összegyűjti az elosztott számítás által generált releváns eseményekről a nyomkövetési információt. Amennyiben hibás szituációt találunk, az elosztott programot újból végrehajthatjuk egy felügyelő mechanizmus vezérlése alatt. A nyomkövetési és visszajátszási technikákat intenzíven kutatták az elmúlt évtizedekben, leginkább a próba-hatásra és nyomkövetéskor keletkező információ mennyiségének csökkentésére fókuszálva. Mindezek ellenére nem minden hibakereső és –javító eszköz támogatja ezt a funkciót. Felhasználói szemszögből a „Nyomkövetés, visszajátszás és hibakeresés” megközelítés komoly hátránya, hogy nem biztosít támogatást más végrehajtási útvonalak vizsgálatához az aktuálisan megfigyeltől kívül.

	Megközelítések/módszerek			
Eszközök	Távoli szekvenciális folyamatok interaktív hibakeresése	Nyomkövetés, visszajátszás és hibakeresés	Integrált tesztelés, aktív kontrol és hibakeresés	Globális prédikátumok automatikus detektálása, aktív kontrol és hibakeresés
DDT [I]	✓	✎ feltételes töréspontok és szinkronizáció	✎ feltételes töréspontok és szinkronizáció	✎ folyamatok közötti összehasonlítás
TotalView [J]	✓	✓ ellenőrzőpont SGI/IRIX és IBM/AIX -en	✎ sorompó és kiértékelési pontok	✎ kiértékelési pontok
MAD [B][C][F]	✓	✓ NOPE segítségével	✓ NOPE segítségével	✘ fejlesztés alatt
DDBG & STEPS [G][H]	✓ DDBG segítségével	✓ STEPS segítségével	✓ STEPS segítségével	✎ kiértékelési funkciók, off-line analízis
P2D2 [D]	✓	✘	✎ vezérlési halmazok	✎ átszabható grid megjelenítő
P-GRADE [E][4]	✓ DIWIDE hibakereső eszközzel	✓ markrólépés motorral	✓ GRSIM (CPN) szimulátorral	✓ TLC temp. logikai ellenőrzővel

Jelmagyarázat: ✓ Teljesen támogatott ✘ Nem támogatott ✎Részlegesen támogatott¹

¹ A felhasználó felelős a felsorolt alapfunkciók helyes használatáért.

Az “*Integrált tesztelés, aktív kontrol és hibakeresés*” módszer megkísérel túllépni a fent említett egyszerű passzív nyomkövetés és visszajátszás korlátain. Hibakeresési és –javítási célokból már számos szerző [L][M][N] javasolt különböző megközelítéseket az elosztott program végrehajtásának aktív vezérlése kapcsán.

Az “*Globális prédikátumok automatikus detektálása, aktív kontrol és hibakeresés*” megközelítés megpróbálja a felhasználó bizonyosságát tovább növelni az alkalmazás megbízhatóságával kapcsolatban. Így az előző megközelítés eredményeire alapozva lehetővé teszi helyességi kritériumok, globális prédikátumok specifikálását. Ezek a globális prédikátumok aztán automatikusan kiértékelésre kerülnek a detektálási algoritmusok segítségével (on-line vagy off-line üzemmódban).

Az előző táblázat összehasonlítja a leginkább kidolgozott hibakeresési és –javítási eszközöket a négy fő hibakeresési és –javítási megközelítések alapján. A táblázat végén a P-GRADE környezetet is megtalálhatjuk. Téziseimben a legfőbb cél, hogy felhasználóbarát és automatizált megoldásokat adjak mind a négy megközelítésre a P-GRADE integrált eszközeinek segítségével: a DIWIDE elosztott hibakereső és –javító eszközzel, a makrólépés motorral, a GRSIM szimulátorral és a TLC temporális logikai ellenőrző motorral. További céloom, hogy a kidolgozott módszereket általánosítsam metaszámítási alkalmazások irányába is.

III. Alkalmazott vizsgálati módszerek

Munkám során első céloom volt, hogy a modellellenőrzés területéről már ismert *formális módszerekkel* bebizonyítsam a P-GRADE környezetben belül a GRAPNEL programok új makrólépés alapú végrehajtásának helyességét.

Az első tézisben, a színezett Petri-háló (CPN) formalizmusát választottam [R][S][T] a GRAPNEL programok *modellelésre* hibakeresési és –javítási szempontból. A CPN-re történő transzformáció a GRAPNEL programok osztályreprezentációján alapul követve annak hierarchikus tervezési koncepcióját. A létrehozott CPN modell XML leírása egy elterjedt CPN szimulációs eszköz formátumát követi.

A makrólépés alapú végrehajtás formális leírása a bevezetett Petri-háló modell állapotterére támaszkodik. Ezután a makrólépés koncepció helyességét formálisan, a Kripke struktúrák részleges rendezésével [O][P][Q] bizonyítom be, melyek egyrészt a CPN modell szabad lefutásainál, másrészt a makrólépés alapú végrehajtásakor megfigyelhető állapotterekből származtathatóak.

Második célként továbbfejlesztettem a makrólépés alapú hibakeresési és –javítási módszert további *modellellenőrző technikákat* [Y] hasznosítva a párhuzamos hibakeresés- és javítás területén. A temporális logikai specifikáció futásidejű kiértékeléséhez [Z] bevezetett támogatáshoz *állapotgép* leírást használtam fel. A Petri-háló *szimulátor* [V][W][X] a makrólépés alapú végrehajtás során képes irányítani és optimalizálni az állapottér bejárását, egy *statikus analízáló* eszköz (egy partícionáló algoritmus) alosztályokba sorolja a folyamatokat, valamint a Rayleigh *hibamodellt* alkalmazva a GRAPNEL programokban lévő hibák sűrűsége megbecsülhetővé válik.

Végezetül, a makrólépés alapú végrehajtást *általánosítottam* a metaszámítási alkalmazások irányában. Az ismertetett megközelítés a HARNESS metaszámítási

keretrendszer újszerű tervezési módszereit követte és egy adaptív, nyitott architektúrát vezettem be a metaszámítási alkalmazások hibakereséséhez és -javításához. Az elérhető hibakereső és -javító eszközök *vizsgálata*, valamint a metaszámítási alkalmazások által támasztott igények *követelményanalízise* után új hibakeresési és -javítási mechanizmusokat fejlesztettem ki a helyi és távoli metódushívások egységesítésére, az egyes folyamatok konzisztens globális állapotainak átvitelére tetszőleges hibakereső és -javító eszközök között, valamint a metaszámítási alkalmazások makrólépés alapú végrehajtásához.

IV. Új tudományos eredmények²

1 Makrólépés alapú hibakeresési és -javítási technika GRAPNEL programokhoz

A P-GRADE fejlesztőkörnyezetben a párhuzamos programok a GRAPNEL hibrid programozási nyelv szintakszisára és szemantikájára alapulva hozhatóak létre. A GRAPNEL különböző nyelvi elemeket biztosít a párhuzamosság, az elosztottság, a konkurencia, valamint különböző hierarchikus szinteken a folyamatok közötti kommunikáció grafikus kifejezésére. Eközben a szekvenciális kód a már meglévő szekvenciális programokból öröklődhet.

Az első tézis célja, hogy formális keretrendszert definiáljon a GRAPNEL programok makrólépés alapú végrehajtásának helyességének a bizonyításához.

1. Új korlátozásokat és nyelvi elemeket vezettem be a GRAPNEL nyelvbe, hogy hibakeresési és -javítási célból automatikusan lehessen generálni hozzá a hierarchikus színezett Petri-háló modellt. Majd színezett Petri-háló minták segítségével leírtam a transzformációs módszereket ennek az új GRAPNEL* nyelvnek minden egyes elemére. A transzformációs lépések alapján megmutattam, hogy *minden GRAPNEL* programhoz létezik Petri-háló model és az legenerálható a hibakeresés és -javítás folyamatához.*

A disszertáció kapcsolódó fejezete: 2.1

Kapcsolódó publikációk: [1][2][5][7][9] (GRAPNEL) [3][22][25][27] (CPN)

2. A GRAPNEL* programok különböző hibakeresési és -javítási ciklusokban megfigyelhető nem-determinisztikus viselkedésnek a kezelésére formálisan megfogalmaztam egy újszerű, az egymást követő konzisztens globális állapotokat automatikusan létrehozó ún. makrólépés alapú végrehajtást. Az állapottér makrólépésekkel történő bejárását (a Végrehajtási Fa generálását) szintén bevezettem, és formális leírása a Petri-háló modell Occurence Gráfján alapul. Megmutattam, hogy *az Occurance Gráfhoz formális bejárési szabályok definiálhatóak az állapotátmenetek reprezentatív halmazának*

² A tézisekhez kapcsolódó állítások félkövér karakterekkel kerültek a leírásba.

kiválasztásával, amely tetszőleges GRAPNEL program esetén, annak makrólépés alapú végrehajtását fogja eredményezni.* Így megmutattam, hogy a szoftverfejlesztő egy olyan végrehajtási mechanizmust alkalmazhat a GRAPNEL* párhuzamos programoknál, ami hasonló a hagyományos szekvenciális programok lépésről-lépésre történő végrehajtásához.

A disszertáció kapcsolódó fejezete: 2.2

Kapcsolódó publikációk: [8][13][27]

3. Ebben az altézisben bebizonyítottam a makrólépés alapú hibakeresési és –javítási módszer helyességét. Első lépésként, mind a Végrehajtási Fát (a makrólépés alapú végrehajtás eredményét), mind az Occurrence Gráfot (a GRAPNEL* teljes állapotterére vonatkozóan) Kripke struktúrává alakítom, így megkapható a KS_e and KS_p struktúrák. Ezt követően bebizonyítom, hogy a *reprezentáns állapotátmeneteket makrólépés alapon kiválasztó algoritmus az egyfajta részleges rendezés a KS_e and KS_p Kripke struktúrákon, és a Kripke struktúrák dadogó (stuttering) ekvivalenciába állnak egymással.* Ezáltal, a felhasználó úgy kap egymást követő globális állapotokat (makrólépéseket), hogy releváns információ nem veszik el a megfigyelt rendszer viselkedését illetően.

A disszertáció kapcsolódó fejezete: 2.3

Kapcsolódó publikációk: [27]

2 Modell ellenőrzési módszerek a párhuzamos programok hibakeresése és –javítása céljából

A temporális logika (TL) és a színezett Petri-háló (CPN) megfelelő megközelítésnek bizonyult egy rendszer (program) dinamikus viselkedésnek leírására, amely több aszinkron módon végrehajtott komponensből (folyamatból) áll.

Ennek a tézisnek a legfőbb célja, hogy az eredeti makrólépés alapú hibakeresési és –javítási módszer hatékonyságát és felhasználhatóságát növelje azáltal, hogy automatikusan összeveti a GRAPNEL* programtól elvárt viselkedést annak megfigyelt viselkedésével. Így az újszerű módszer mind a párhuzamos hibakeresési és –javítási módszerek, mind a modellellenőrzési algoritmusok területéről ötvöz különböző koncepciókat.

1. A bemutatott elméleti háttérre támaszkodva bemutatok egy módszert a hibakeresési és –javítási keretrendszer integrációja kapcsán, ahol az aktuális program lefutásai, amit a makrólépés alapú hibakereső és –javító eszköz vezérel, egyben egy univerzum, ahol a felhasználó által definiált temporális logikai formulák ellenőrzése történik. Bemutatom a keretrendszer inicializálási fázisát, valamint annak a módját, ahogy beilleszthetőek, illetve detektálhatóak

a futásidejű temporális logikai állítások. Ismertetem a temporális logikai formulákból hivatkozott atomi prédikátumok kiértékeléséhez kidolgozott támogatást, valamint egy állapotgép segítségével a kommunikációs protokollt egy általános célú TL ellenőrzővel. Ezekre az eredményekre támaszkodva megmutatom, hogy a **makrólépés alapú végrehajtás közben a temporális logikai kifejezések egy osztálya (LTL_x) kiértékelhetővé válik a Végrehajtási Fa útvonalain**. Ily módon, a temporális logikai állításoknak köszönhetően radikálisan csökkenthető a felhasználótól szükséges beavatkozások száma a hibák felismeréséhez.

A disszertáció kapcsolódó fejezete: 3.1

Kapcsolódó publikációk: [14][25][27][33][34]

2. Ebben a tézisben a GRAPNEL* programokhoz tartozó CPN modellre támaszkodva ismertetem a színezett Petri-háló (CPN) szimulációs motor egy lehetséges integrációs módját a hibakeresési és -javítási keretrendszerbe. Bemutatom, hogy a **CPN szimulációs motor képes az állapotér makrólépés alapú bejárását** (a Végrehajtási Fa felépítését) **a hibás szituációk irányába terelni, valamint felismerni a már korábban bejárt végrehajtási útvonalakat**. A bemutatott módszer szimulációs és irányítási technikáival segíthet a felhasználónak a hibák felderítésében.

Munkám részeként a következő technikákat vizsgáltam még meg, hogy tovább javítsam a keretrendszert. (1) Csökkentett CPN modellt, ha a GRAPNEL program nem elégíti ki a korlátozásokat. (2) Néhány módszert a GRAPNEL programok determinisztikus és nem-determinisztikus részekre bontására, hogy még hatékonyabbá váljon a hibakeresés és -javítás. (3) Elosztott konfigurációt, ahol több végrehajtási útvonalat lehet szimultán módon futtatni. (4) Egy analízator eszközt a hibasűrűség becslésére szolgáló Rayleigh modellre alapozva, ahol az analízator képes javaslatot tenni a párhuzamos szoftver kiadására, amikor a program megbízhatósága elérte a megkívánt szintet. (5) Egy módszert, ami bizonyos esetekben képes növelni a kiadott program megbízhatóságát azáltal, hogy a már tesztelt állapotéren belülré kényszeríti annak állapotait.

A disszertáció kapcsolódó fejezete: 3.2

Kapcsolódó publikációk: [25][27]

3 Hibakeresés és -javítás metaszámítási alkalmazásokban

Az újonnan megjelenő nagy számítási teljesítményt igénylő (HPC) alkalmazásokhoz szükségessé tették a heterogén és földrajzilag elosztott erőforrások kihasználását is. Ezek az alkalmazások különböző típusú hálózatokat használnak fel a szuperszámítógépek, a nagy adatbázisok, a vizualizációs berendezések és a tudományos eszközök integrálásához, hogy ún. hálózatba kapcsolt virtuális szuperszámítógépeket vagy *metaszámítógépeket* alkossanak, melyek egyszerű asztali számítógépekről is elérhetőek. Amíg a metaszámítási rendszerek felépítéséhez szükséges fizikai infrastruktúra egyre elterjedtebbé válik,

addig a metaszámítási környezet heterogén és dinamikus természete új kihívásokat támaszt az alkalmazásfejlesztő és hibakereső és –javító eszközök felé.

Ennek a tézisnek a legfontosabb célja, hogy általánosítsa a makrólépés alapú hibakeresési és –javítási módszereket a metaszámítási alkalmazásokra.

1. Ebben az altézisben *egy adaptív hibakeresési és –javítási keretrendszert (metadebugger) mutatok be HARNESS metaszámítási alkalmazásokhoz, ami* követi az ún. szoftver hátlap megközelítést, így *képes kezelni a hibakeresési és –javítási fázisban a metaszámítási környezet dinamikus viselkedését*. A metahibakereső és –javító eszköz képes exportálni és importálni különböző hibakereső és –javító eszközök között a többszálás alkalmazások folyamatainak konzisztens állapotait, így a heterogén környezetben elérhetővé válik a harmadik fél által kifejlesztett hibakereső és –javító eszközök meghívása is. Számos új tulajdonságot vezettem be a metahibakereső és –javító eszközbe, magában foglalva az automatikus kontextus menedzselő („*step into*”) mechanizmust a távoli metódushívásokhoz (RMI), ami egységesíti a lokális és távoli metódushívások hibakeresését és –javítását még akkor is, ha az alkalmazás egy metaszámítási platformon kerül végrehajtásra. Erre az RMI támogatásra alapozva kidolgoztam egy monitorozó és vizualizációs alrendszert a metaszámítási alkalmazásokhoz.

A disszertáció kapcsolódó fejezete: 4.1

Kapcsolódó publikációk: [11][13][28][29]

2. Ez az altézis a metaszámítási alkalmazások nem-determinisztikus viselkedésére és architektúra függőségére fókuszál hibakeresési és –javítási szempontból. Megmutatom, hogy ebből a célból, *a makrólépés alapú hibakeresési és –javítási módszerek általánosíthatóak HARNESS metaszámítási alkalmazásokra*. Az új módszer módosított kollektív töréspontokon és makrólépéseken, valamint erőforrás-transzlációs táblákon alapul, de ehhez korlátozásokat kellett bevezetnem a metaszámítási alkalmazásban használható kommunikációs lehetőségek tekintetében. A metahibakereső és –javító keretrendszer architektúráját továbbfejlesztettem egy makrólépés vezérlő modullal, ami a metahibakereső és –javító keretrendszer monitorozási képességeire támaszkodik. Egy erőforrás-transzlációs mechanizmust dolgoztam ki a végrehajtási környezettel kapcsolatos reprodukálási problémák kiküszöbölése érdekében, valamint egy algoritmust a metaszámítási alkalmazások környezetfüggőségének a tesztelésére.

A disszertáció kapcsolódó fejezete: 4.2

Kapcsolódó publikációk: [10][28]

V. Az eredmények alkalmazása

A tézisekben bemutatott eredményeket számos konferencia kiadványban és folyóiratban publikáltam, valamint nemzetközi tudományos fórumokon és kiállításokon is bemutatásra kerültek.

A P-GRADE fejlesztőkörnyezet egyre inkább felkelti az európai és amerikai egyetemek és az IT cégek figyelmét. A P-GRADE rendszert sikeresen alkalmazták a meteorológiában egy ultrarövidtávú időjárás előrejelző rendszer párhuzamosítására (Országos Meteorológiai Szolgálat) [3][12][15], a mérnöki tudományok területén városi forgalom szimulációra (University of Westminster), valamint reakció-diffúzió rendszerek modellezésére a kémiában (ELTE) [23][26][30].

A bemutatott tudományos eredmények többsége már megvalósításra került a P-GRADE környezetben és a szoftverfejlesztők kihasználhatják a bemutatott hibakeresési és -javítási módszerek által nyújtott előnyöket, hogy a szoftvertermékek megbízhatóságát növeljék. A közelmúltban a P-GRADE rendszer továbbfejlesztésre került, hogy támogassa mind a zökkenőmentes alkalmazás migrációt a hagyományos párhuzamos és elosztott platformokról a Grid környezetekre [17][20], mind a munkafolyam (*workflow*) alapú komplex alkalmazásokat [4][16] [18][21][24].

Ezáltal a P-GRADE eszközeivel megtervezett és a hibakeresésen/javításon átesett alkalmazások ezeken az új platformokon is üzembe helyezhetővé váltak, így biztosítva még több lehetőséget a végfelhasználók számára.

A metaszámítási alkalmazásokhoz kapcsolódó eredmények más metaszámítási vagy Grid számítási keretrendszerben is alkalmazhatóak (pl. [19]), melyek a HPC és az adat-intenzív alkalmazások egyre inkább elfogadott, standard platformjává válnak.

VI. Publikációs lista

Folyóiratcikkek

- [1] P. Kacsuk, G. Dózsa, T. Fadgyas and R. Lovas: *GRADE: a Graphical Programming Environment for Multicomputers*, Journal of Computers and Artificial Intelligence, Slovak Academy of Sciences, Vol. 17, No. 5, pp. 417-427, 1998
- [2] P. Kacsuk, G. Dózsa, T. Fadgyas and R. Lovas: *The GRED Graphical Editor for the GRADE Parallel Program Development Environment*, Journal of Future Generation Computer Systems, Vol. 15, No. 3, pp. 443-452, Elsevier Science, 1999
- [3] R. Lovas, P. Kacsuk, A. Horvath, A. Horanyi: *Application of P-GRADE Development Environment in Meteorology*, Distributed and Parallel Systems. Special issue of Scalable Computing: Practice and Experience. Vol. 6, No. 2, pp. 13-22. 2005 July (ISSN 1895-1767)

- [4] P. Kacsuk, G. Dózsa, J. Kovács, R. Lovas, N. Podhorszki, Z. Balaton, G. Gombás: *P-GRADE: a Grid Programming Environment*, Journal of Grid Computing, Volume 1, Issue 2, 2004, Pages 171 - 197

Könyvfejezet

- [5] P. Kacsuk, G. Dózsa, R. Lovas: *The GRADE Graphical Parallel Programming Environment*, In the book: *Parallel Program Development for Cluster Computing: Methodology, Tools and Integrated Environments* (Chapter 10), Editors: P. Kacsuk, J.C. Cunha and S.C. Winter, pp. 231-247, Nova Science Publishers New York, 2001

Konferenciacikkek

- [6] A. Bäcker, D. Ahr, O. Krämer-Fuhrmann, R. Lovas, H. Mierendorff, H. Schwamborn, J. G. Silva, K. Wolf: *WINPAR, Windows-Based Parallel Computing*, In: *Parallel Computing: Fundamentals, Applications and New Directions*, Series of Advances in Parallel Computing, Vol. 12, pp. 495-502, Elsevier Science, 1998
- [7] P. Kacsuk, G. Dózsa, T. Fadgyas and R. Lovas: *The GRED Graphical Editor for the GRADE Parallel Program Development Environment*, In: *High-Performance Computing and Networking*, Lecture Notes in Computer Science, Vol. 1401, pp. 728-737, Springer Verlag, 1998
- [8] P. Kacsuk, R. Lovas, J. Kovács: *Systematic Debugging of Parallel Programs in DIWIDE Based on Collective Breakpoints and Macrosteps*, In: *EuroPar '99 Parallel Processing*, Lecture Notes in Computer Science, Vol. 1685, pp. 90-97, Springer-Verlag, 1999
- [9] G. Dózsa, D. Drótos, R. Lovas: *Translation of a High-Level Graphical Code to Message-Passing Primitives in the GRADE Programming Environment*, In: *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, Lecture Notes in Computer Science, Vol. 1908, pp. 258-265, Springer-Verlag, 2000
- [10] R. Lovas, V. Sunderam: *Extension of macrostep debugging methodology towards metacomputing applications*, In: *Computational Science - ICCS 2001*, Lecture Notes in Computer Science, Vol. 2074, p. 263-272, Springer Verlag, 2001
- [11] R. Lovas, V. Sunderam: *A Metadebugger Prototype for the HARNCESS Metacomputing Framework*, Proc. of the Tenth IEEE International Symposium on High Performance Distributed Computing, pp. 427-428, San Francisco, CA, USA, 2001
- [12] Lovas R., Horváth Á.: *Ultrarövidtávú meteorológiai előrejelző rendszer párhuzamosítása a P-GRADE fejlesztőkörnyezettel*, Proc. of NETWORKSHOP '2002, pp. 56 + CD-ROM, Eger, Hungary, 2002
- [13] R. Lovas, V. Sunderam: *Debugging of Metacomputing Applications*, Proc. of the 16th International Parallel and Distributed Processing Symposium (IPDPS-JPDC), pp. 119 + CD-ROM, Fort Lauderdale, FL, USA, 2002
- [14] J. Kovacs, G. Kusper, R. Lovas, W. Shreiner: *Integrating Temporal Assertions into a Parallel Debugger*, In: *EuroPar 2002 Parallel Processing*, Lecture Notes in Computer Science, vol. 2400, pp. 113-120, Springer-Verlag, 2002
- [15] R. Lovas, P. Kacsuk, A. Horvath, A. Horanyi: *Application of P-GRADE Development Environment in Meteorology*, In: *Distributed and Parallel Systems, Cluster and Grid Computing*, pp. 109-116, Kluwer Academic Publishers, 2002

- [16] P. Kacsuk, R. Lovas, J. Kovacs, G. Dozsa, N. Podhorszki: *Metacomputing Support by P-GRADE*, GGF8 Workshop on Grid Applications and Programming Tools, 2003
- [17] P. Kacsuk, R. Lovas, J. Kovács, F. Szalai, G. Gombás, N. Podhorszki, A. Horváth, A. Horányi, I. Szeberényi, T. Delaitre, G. Terstyánszky, A. Gourgoulis: *Demonstration of P-GRADE job-mode for the Grid*, In: Euro-Par 2003 Parallel Processing, Lecture Notes in Computer Science, Vol. 2790, pp. 1281-1286, Springer-Verlag, 2003
- [18] G. Dozsa, P. Kacsuk, Sz. Illes, Cs. Nemeth, Gy. Rabai, Z. Farkas, G. Gombas, R. Lovas: *Constructing and executing Grid workflow applications by Grid portal technology*, IEEE International Conference on Cluster Computing, pp. 19-22, Hong Kong, 2003
- [19] Z. Juhasz, R. Lovas, and P. Kacsuk: *JGrid: A Jini-based Service Grid*, IEEE International Conference on Cluster Computing, pp. 28-30, Hong Kong, 2003
- [20] R. Lovas, J. Kovacs, G. Gombas, N. Podhorszki, Z. Balaton, P. Kacsuk, I. Szeberenyi, T. Delaitre, and A. Gourgoulis: *Migration and monitoring of P-GRADE parallel jobs in the Grid*, IEEE International Conference on Cluster Computing, pp. 8-11, Hong Kong, 2003
- [21] R. Lovas, G. Dózsa, P. Kacsuk, N. Podhorszki, D. Drótos: *Workflow Support for Complex Grid Applications: Integrated and Portal Solutions*, In: Grid Computing – Second European AcrossGrids Conference, AxGrids 2004, Nicosia, Cyprus, Lecture Notes in Computer Science, Vol. 3165, pp. 129-138, Springer-Verlag, 2004
- [22] B. Vécsei, R. Lovas: *Debugging method for parallel programs based on Petri-net representation*, Proceedings of MicroCAD 2004, Miskolc, Hungary, pp. 413-420, 2004
- [23] R. Lovas, P. Kacsuk, I. Lagzi, T. Turányi: *Unified development solution for cluster and grid computing and its application in chemistry*, In: Computational Science and Its Applications – ICCSA 2004: International Conference, Assisi, Italy, Lecture Notes in Computer Science, Vol. 3044, pp. 226-235, Springer-Verlag, 2004
- [24] Cs. Németh, G. Dózsa, R. Lovas and P. Kacsuk: *The P-GRADE Grid portal*, In: Computational Science and Its Applications – ICCSA 2004: International Conference, Assisi, Italy, Lecture Notes in Computer Science, Vol. 3044, pp. 10-19, Springer-Verlag, 2004
- [25] R. Lovas, B. Vécsei: *Integration of formal verification and debugging methods in P-GRADE environment*, In: Distributed and Parallel Systems: Cluster and Grid Computing, Kluwer International Series in Engineering and Computer Science, Vol. 777, pp. 83-92, 2004
- [26] I. Lagzi, R. Lovas, T. Turányi: *Development of a grid enabled chemistry application*, In: Distributed and Parallel Systems: Cluster and Grid Computing, Kluwer International Series in Engineering and Computer Science, Vol. 777, pp. 137-144, 2004

Konferencia előadások és poszterek

- [27] R. Lovas, P. Kacsuk: *Enhanced Macrostep-based Debugging Methodology for Parallel Programs*, The Third Conference of PhD Students in Computer Science, pp. 72, Szeged, Hungary, 2002 (honored with *Excellent Talk Award*)
- [28] R. Lovas: *A Debugger for Metacomputing Applications*, 2nd US/Hungarian Workshop on Cluster Computing, Metacomputing and Grid Computing, Budapest, February 6, 2002

- [29] R. Lovas: *Debugging and Visualization in the Harness Metacomputing Framework*, 1st US/Hungarian Workshop on Cluster Computing, Metacomputing and Grid Computing, Madison, Wisconsin, USA, March 15, 2001
- [30] T. Turányi, F. Izsák, R. Lovas, P. Kacsuk: *Parallelisation of an algorithm for reaction-diffusion systems applying P-GRADE environment*, Workshop 5 of the ESF Programme “REACTOR”, Prague, Czech Republic, 2004

Nem lektorált konferenciák

- [31] P. Kacsuk, G. Dózsa, R. Lovas, T. Fadgyas: *Enhancing GRADE towards a professional parallel programming environment*, Proc. of the 3rd Workshop of Stimulation of European Industry Through High Performance Computing, Madrid, 1998

Technikai riportok

- [32] Gabor Dozsa, Robert Lovas and Peter Kacsuk: *A Target Set of Parallel Architectures for the Design of the Semi-Automatic Urine Analyser*, <http://www.cpc.wmin.ac.uk/~ahmed/reports.html>, AHMED/3 project report, October 1997.
- [33] Gabor Kusper, Wolfgang Schreiner, Robert Lovas: *Integrating Temporal Specifications as Runtime Assertions into Parallel Debugging Tools*, RISC-Linz Report Series No. 02-07, <http://www.risc.uni-linz.ac.at/library/>, March 2002
- [34] Jozsef Kovacs, Gabor Kusper, Robert Lovas, Wolfgang Schreiner: *Integrating Temporal Assertions into a Parallel Debugger*, RISC-Linz Report Series No. 02-12, <http://www.risc.uni-linz.ac.at/library/>, May 2002

VII. REFERENCIÁK

- [A] José C. Cunha, João Lourenço, Vitor Duarte: *Debugging of Parallel and Distributed Programs*. In the book: *Parallel Program Development for Cluster Computing: Methodology, Tools and Integrated Environments* (Chapter 5), pp. 101-136, Nova Science Publishers New York, 2001
- [B] Dieter Kranzlmüller, Axel Rinnac: *Parallel Program Debugging with MAD - A Practical Approach*. International Conference on Computational Science 2003, pp. 201-212
- [C] D. Kranzlmüller and J. Volkert. NOPE: A Nondeterministic Program Evaluator. In P. Zinterhof et al., editors, *Parallel Computation, Proceedings of ACPC'99*, 4th International ACPC Conference, volume 1557 of Lecture Notes in Computer Science, pages 490-499, Salzburg, Austria, February 16-18, 1999. Springer, Berlin.
- [D] Robert Hood. The p2d2 project: building a portable distributed debugger. Proceedings of the SIGMETRICS symposium on Parallel and distributed tools, May 22 - 23, 1996, Philadelphia, PA USA
- [E] J. Kovacs, P. Kacsuk. The DIWIDE Distributed Debugger on Windows NT and UNIX Platforms, *Distributed and Parallel Systems, From Instruction Parallelism to Cluster Computing*, Eds.: P. Kacsuk and G. Kotsis, Cluwer Academic Publishers, 2000.
- [F] D. Kranzlmüller, S. Grabner, J. Volkert. Event Graph Visualization for Debugging Large Applications. Proc. SPDT'96, ACM SIGMETRICS Symp. on Parallel and Distr. Tools, Philadelphia, USA, pp. 108-117, 1996
- [G] Henryk Krawczyk, Piotr Kuzora, Marcin Neyman, Jerzy Proficz and Bogdan Wiszniewski: *STEPS - a Tool for Structural Testing of Parallel Software*. In the book: *Parallel Program Development for Cluster Computing: Methodology, Tools and Integrated Environments* (Chapter 16), pp. 334-354, Nova Science Publishers New York, 2001
- [H] José C. Cunha, João Lourenço and Vitor Duarte: *The DDBG Distributed Debugger*. In the book: *Parallel Program Development for Cluster Computing: Methodology, Tools and*

- Integrated Environments (Chapter 13), pp. 292-303, Nova Science Publishers New York, 2001
- [I] Allinea Software Ltd.: Distributed Debugger Tool v1.8, User Guide, 2004
 - [J] Etnus, LLC. TotalView debugger. Available online at <http://www.etnus.com/TotalView/MPL.html>
 - [K] M. Migliardi, V. Sunderam, A. Geist, J. Dongarra. Dynamic Reconfiguration and Virtual Machine Management in the Harness Metacomputing System, Proc. of ISCOPE98, pp. 127-134, Santa Fe', New Mexico (USA), December 8-11, 1998.
 - [L] A. Tarafdar and V. K. Garg: Predicate control for active debugging of distributed programs, Proceedings of the 1st Merged International Parallel Processing Symposium and Symposium on Parallel and Distributed Processing (IPPS/SPDP-98), pages 763-769, Los Alamitos, March 30-April 3 1998
 - [M] João Lourenço, José C. Cunha: Fiddle: A Flexible Distributed Debugging Architecture, ICCS 2001, San Francisco, CA, USA, 2001, pp. 821-830
 - [N] Frey and Oberhuber, M.: Testing and Debugging Parallel and Distributed Programs with Temporal Logic Specifications, Proc. of Second Workshop on Parallel and Distributed Software Engineering 1997, pages 62-72, Boston, May 1997
 - [O] Alberto Lluch-Lafuente, Stefan Leue and Stefan Edelkamp: Partial Order Reduction in Directed Model Checking, In: Proceedings of the 9th International SPIN Workshop on Model Checking Software, Springer LNCS, Grenoble, April 2002
 - [P] E. M. Clarke, O. Grumberg, M. Minea, D. Peled. State space reduction using partial order techniques. Software Tools for Technology Transfer, vol. 3, no. 1, Springer Verlag, 1999, pp. 279-287.
 - [Q] R. Kurshan, V. Levin, M. Minea, D. Peled, H. Yenigün. Static partial order reduction. Proceedings of the 4th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Lisbon, Portugal, March/April 1998, pp. 345-357
 - [R] K. Jensen: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Volume 1, Basic Concepts. Monographs in Theoretical Computer Science, Springer-Verlag, 1992
 - [S] K. Jensen: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Volume 2, Analysis Methods. Monographs in Theoretical Computer Science, Springer-Verlag, 1994
 - [T] M. Software "Desgin/CPN. A Tool Package Supporting the Use of Colored Petri Nets" Tech. Rep., Meta Software Corporation, Cambridge, MA, USA, 1991
 - [U] P. Rondogiannis, M.H.M Cheng. Petri-net-based deadlock analysis of Process Algebra programs. Science of Computer Programming, 1994. Vol. 23 (1), pp. 55-89
 - [V] Matthew B. Dwyer, Lori A. Clarke, and Kari A. Nies. A compact petri net representation for concurrent programs. Technical Report TR 94-46, University of Massachusetts, Amherst, 1994
 - [W] Jim Greene: Ensuring Delivery of Highly Reliable, Complex Software Releases, QSM White Paper, October 2003
 - [X] I. Majzik, A. Pataricza and A. Bondavalli: Stochastic Dependability Analysis of System Architecture Based on UML Models. In R. de Lemos, C. Gacek and A. Romanovsky (eds.): Architecting Dependable Systems, LNCS-2667, Springer Verlag, Berlin, 2003, pp 219-244
 - [Y] E. M. Clarke, Jr., O. Grumberg, and D. A. Peled. Model Checking. MIT Press, Cambridge, MA, 1999.
 - [Z] Z. Manna and A. Pnueli. The Temporal Logic of Reactive and Concurrent Systems Specification. Springer, Berlin, 1992.
 - [AA] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, B. Mancheck and V. Sunderam. PVM: Parallel Virtual Machine a User's Guide and Tutorial for Networked Parallel Computing, MIT Press, Cambridge, MA, 1994.