Közösségi erőforrás-megosztás alapú számítási modell alkalmazása a gyakorlatban – SZTAKI Desktop Grid –

KORNAFELD ÁDÁM

MTA-SZTAKI, Párhuzamos és Elosztott Rendszerek Kutatólaboratórium
kadam@sztaki.hu

Lektorált

Kulcsszavak: dc-api, boinc, hierarchikus desktopgrid, közösségi erőforrás-megosztás alapú számítási modell, SZTAKI Desktop Grid

Számításigényes feladatok megoldásához a szuperszámítógépek mellett különböző grid technológiák is sikerrel alkalmazhatóak. Ezen technológiák kifejlesztésének fő célja a költségcsökkentés mellett az volt, hogy bárki felajánlhassa erőforrását a grid számára és bárki felhasználója lehessen a grid erőforrásainak. Ez az elképzelés azonban a mai napig nem valósult meg teljes mértékben. A grid technológiák fejlődése két külön úton indult meg. A clustergrid modellre épülő gridekben bárki hozzáférhet felhasználóként a grid erőforrásaihoz, de erőforrás csak megfelelő szakértelem birtokában ajánlható fel. Ezzel szemben a közösségi erőforrás-felajánláson alapuló számítási modellt alkalmazó gridekhez bárki felajánlhatja erőforrását, de a grid erőforrásait csak az azt felállító intézmény használhatja. Ez utóbbi modell némi módosítással mégis alkalmas lehet a grid technológiák eredeti koncepciójának megvalósítására. Jelen cikk a közösségi erőforrás-felajánláson alapuló számítási modell kiterjesztésének lehetőségét mutatja be a SZTAKI Desktop Griden keresztül.

1. Bevezetés

Napjaink tudományos kutatásai során gyakran merül fel igényként hatalmas mennyiségű információ feldolgozása, mely számítógépek segítsége nélkül szinte elképzelhetetlen. Olykor a rendelkezésre álló idő rövidsége teszi szükségessé minél nagyobb számítási teljesítmény alkalmazását. Remek példa erre az időjárás-előrejelzés, ahol a rendelkezésre álló mérési adatokból még azelőtt kell a jövőre nézve releváns következtetéseket levonni, mielőtt azok idejétmúlttá válnának.

A különböző grid technológiák az ilyen és ehhez hasonló igények kiszolgálását tűzték ki célul. Az eredeti elképzelés – miszerint bárki felajánlhat erőforrást a grid számára, valamint az igényeknek megfelelően bárki dinamikusan igényelhet erőforrást a gridtől – azonban nem valósult meg teljes mértékben. Napjainkban két jellemző irányvonalat figyelhetünk meg a grid technológiák fejlődésében. Az egyik irányvonal a clustergrid modellt alkalmazza, míg a másik elképzelés a Közösségi Erőforrás-felajánláson alapuló számítási modellre épülő desktopgrid architektúrát részesíti előnyben. Cikkünkben a desktopgrid architektúrát, annak kialakulását, a benne rejlő lehetőségeket és egy konkrét megvalósítását mutatjuk be.

2. A desktopgrid kialakulása

Számításigényes feladatok megoldásához kutatók már az 1970-es évek elejétől szuperszámítógépek kifejlesztésében látták a megoldást. A kifejlesztett szuperszámítógépek általában be is váltották a hozzájuk fűzött reményeket teljesítmény szempontjából, azonban mind kifejlesztésük, mind pedig üzemeltetésük rendkívül költségesnek bizonyult. Ennek megfelelően mind a mai napig nem teljesen triviális hozzáférni egy szuperszámítógép erőforrásaihoz. Elsősorban költségcsökkentés céljából felmerült az igény egy olyan architektúra kifejlesztésére, mellyel a nagy méreteket öltő költségek jelentős mértékben csökkenthetők, ugyanakkor a kifejlesztett rendszer számítási kapacitása továbbra is versenyképes marad a szuperszámítógépek körében. Ez az elképzelés vezetett a clustergrid architektúra kifejlesztéséhez.

A cél egy olyan grid szolgáltatás kifejlesztése volt, melyhez sok felhasználó hozzá tud férni. Ezt a grid szolgáltatást erőforrások hálózatba kapcsolásával hozták létre. Ahhoz, hogy egy erőforrás a grid részévé válhasson, egy előre meghatározott programcsomagot, úgynevezett middleware-t kell rá telepíteni. Egy ilyen middleware azonban annyira összetett, bonyolult rendszert alkot, hogy az üzemeltetés komoly szakértelmet igényel. Emiatt természetesen adódott, hogy magánszemélyek nem ajánlanak fel erőforrást, csak nagyobb intézmények (pl. egyetemi tanszékek), ahol egy szakértőkből álló csapat felügyeli a komplex, hardware és software komponensekből álló rendszert, ezáltal garantálva annak nagy rendelkezésre állását. Ilyen grid infrastruktúrákra példa a legnagyobb európai grid, az EGEE (Enabling Grids for E-SciencE [1]), és annak magyar virtuális szervezete a HunGrid. Ezekhez a gridekhez felhasználóként bárki hozzáférhet, aki rendelkezik megfelelő felhasználói tanúsítvánnyal, mely tanúsítványkezelő hatóságoknál igényelhető.

A kialakult clustergrid modell a fentebb tárgyalt okok miatt azonban nem alkalmas annak az igénynek a kiszolgálására, hogy bárki felajánlhassa erőforrását a grid számára. Bár a clustergrid modell alkalmazására összeállt intézmények számottevő erőforrásparkkal rendelkeznek, egyértelmű, hogy még nagyobb számítási teljesítmény összefogására nyílik lehetőségünk, ha le-

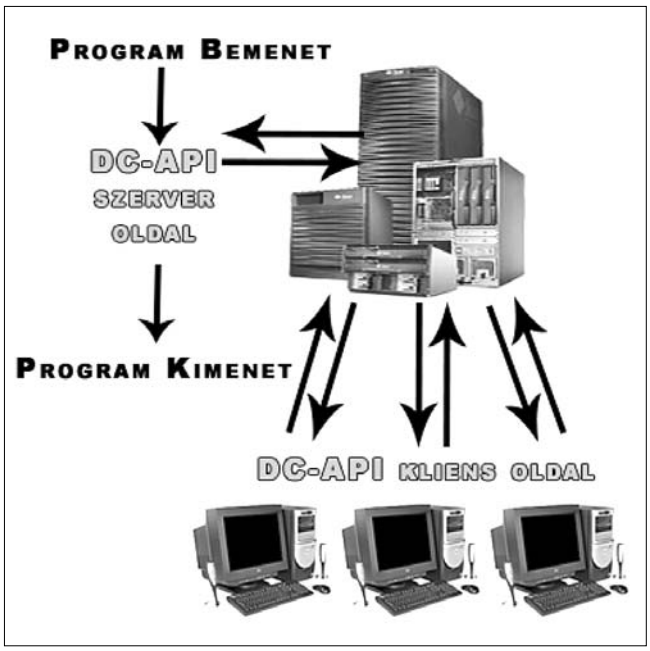
hetővé tesszük, hogy bárki és bármekkora erőforrást felajánlhasson a grid számára. Ez az elképzelés vezetett a Közösségi Erőforrás-felajánláson alapuló számítási modell kialakulásához, melynek sematikus rajza az *1. ábrán* látható.

A modellben a legkisebb építőelem a személyi számítógép (Desktop Computer). Erre utalva kapta a desktopgrid nevet a modellre épülő architektúra.

3. A desktopgrid architektúra

A legismertebb példa, a legelső desktopgrid architektúrájú grid a mind a mai napig futó SETI@Home [2]. A desktopgridek személyi számítógépektől kezdve a legkülönbözőbb erőforrásokat kapcsolnak össze egy központi szerver segítségével, jelentős számítási teljesítményre szert téve ez által. A saját erőforrását donorként felajánló számítógép tulajdonosnak mindössze egyetlen kliens programot kell telepítenie számítógépére, melynek a grid honlapján való rövid regisztrációt követően kapott címet megadva a felajánlott erőforrás máris a desktopgrid szerves részét képezi. A résztvevők számítógépein futó kliens program nem igényel semmilyen felhasználói beavatkozást az adott erőforrás felhasználójától. Ezáltal elértük, hogy bárki hozzájárulhat a grid teljesítményének növeléséhez saját erőforrásának felajánlásával. Ahhoz azonban, hogy sok felajánlott erőforrásra tegyünk szert, szükség van a desktopgrid és elsősorban az azon futatott alkalmazás reklámozására, népszerűsítésére. Emiatt a clustergridekkel ellentétben a desktopgridek nem sok felhasználósak, jellemzően egyetlen programmal egy adott problémára keresik a megoldást. Így általában a desktopgrid rendszert felállító intézmény egyben egyetlen felhasználója is a desktopgridnek.

1. ábra
Közösségi erőforrás-felajánláson alapuló számítási modell



A desktopgrides környezet jellemzően 'mester-szolga' mintán alapuló problémamegoldást jelent. A megoldandó feladatot kis részfeladatokra osztják (például a program bemenetét képező adathalmaz feldarabolásával), melyek egymástól függetlenül, konkurensen végrehajthatóak. A részfeladatokat ezután a desktopgridhez csatlakozó szuverén erőforrások dolgozzák fel. A desktopgrid központi szerverén futó 'mester' program gondoskodik a feladatok szétosztásáról és a 'szolgák' által szolgáltatott részeredmények feldolgozásáról.

A rendszer legnagyobb előnye az egyszerűsége, míg hátránya, hogy csak a 'mester-szolga' paradigmán alapuló programokat tud kiszolgálni. Az architektúra már világszerte bizonyított olyan nagyszabású projektekben, mint a földönkívüli intelligencia nyomait kutató SET@Home, a Föld klímaváltozását kutató Climateprediction@Home, vagy az emberi kórokozókat kutató World Community Grid.

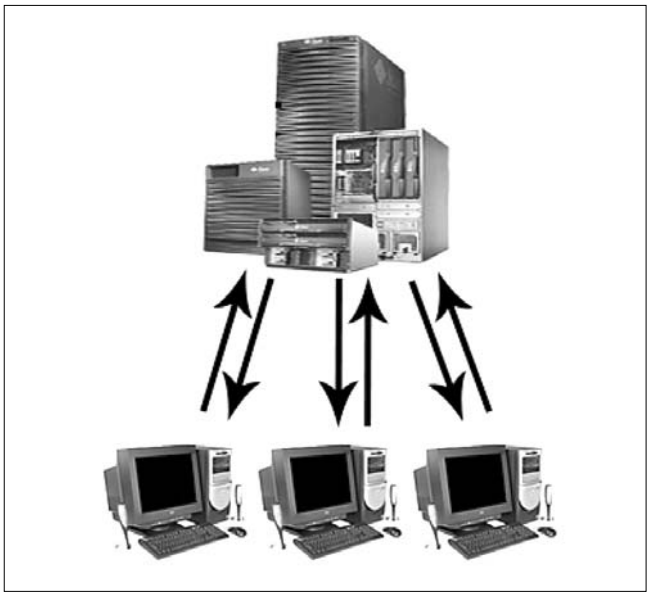
Természetesen a desktopgrid architektúrát eredményesen alkalmazhatjuk kisebb méretekben is. Hasonlóan ahhoz, mint ahogy a személyi számítógép építőeleme a desktopgridnek, egy szinttel feljebb kisebb méretű desktopgridek építőelemei lehetnek egy nagyobb gridnek. Ez egy új koncepció, mellyel közelebb hozhatjuk egymáshoz a grid technológiák két fő irányvonalát, konvergálva ez által a grid technológia eredeti elképzeléséhez. Önálló intézmények könnyűszerrel összekapcsolhatják már meglévő erőforrásparkjukat egy úgynevezett lokális desktopgridbe. A technológia könnyű, felhasználóbarát telepíthetősége miatt a desktopgrid architektúra könnyebben és gyorsabban terjedhet el, mint a robosztusabb architektúrák. Mindemellett, ha lehetővé válik, hogy a lokális desktopgridek megoszthatják egymás között erőforrásaikat a sokkfelhasználós modell előnyei is kihasználhatóak.

Ahhoz, hogy a desktopgridek együttműködhessenek a támogatott architektúrák közé kell emelni a clustereket, meg kell valósítani a desktopgridek többszintű hierarchiáját egy nagy szervezeten belül, valamint az erőforrás-megosztást a különböző desktopgridek között.

4. SZTAKI Desktop Grid

Laborunkban felállított SZTAKI Desktop Grid, önálló desktopgridként, az első építőköve a grid technológiák két nagy trendjének összeházasításában végzett munkánknak. A grid alapját a nyílt forráskódú BOINC (Berkeley Open Infrastructure for Network Computing [3]) platform szolgáltatja. A BOINC platform kifejlesztésével a Berkeley-egyetem kutatói egy olyan nyílt infrastruktúra létrehozását tűzték ki célul, mely alapjául szolgálhat számos olyan nagyszabású tudományos projektnek, melyben személyi számítógépek millióit használják információfeldolgozásra.

A platform két fő részből áll: egy központi szerverből és a hozzá kapcsolódó kliensekből *(2. ábra* – lásd a következő oldalon).



2. ábra A BOINC platform

A szerver komponensekből épül fel, ami lehetővé teszi, hogy az adott komponensek különálló hardveren futtassuk, megfelelő robusztussággal ruházva fel ez által a szervert. Feladata, hogy az adott projekt által szolgáltatott feldolgozandó információt szétdarabolva és adatbázisba rendezve elérhetővé tegye azt a csatlakozó kliensek számára, továbbá a kliensek által feldolgozott és visszaküldött eredményeket hitelesítse és rendszerezze. A számítási kapacitásukat felajánlóknak nincs más dolguk, mint letölteni és telepíteni egy kliens programot számítógépükre.

Attól a ponttól kezdve, hogy az erőforrás felajánlója megadta, hogy milyen beállítások mellett engedélyezi a projekt számára a processzor üres idejének kihasználását, a kliens program teljes mértékben automatizáltan csatlakozva a szerverhez, letölti a projekt által feldolgozandó bemenet egy kis részletét, valamint annak feldolgozásához szükséges programot egy munkacsomag formájában. A program segítségével a kliens megkezdi a kapott bemenet feldolgozását. Amennyiben végzett a feldolgozással, a kapott eredményt visszatölti a szerverre, és újabb munkát kér attól.

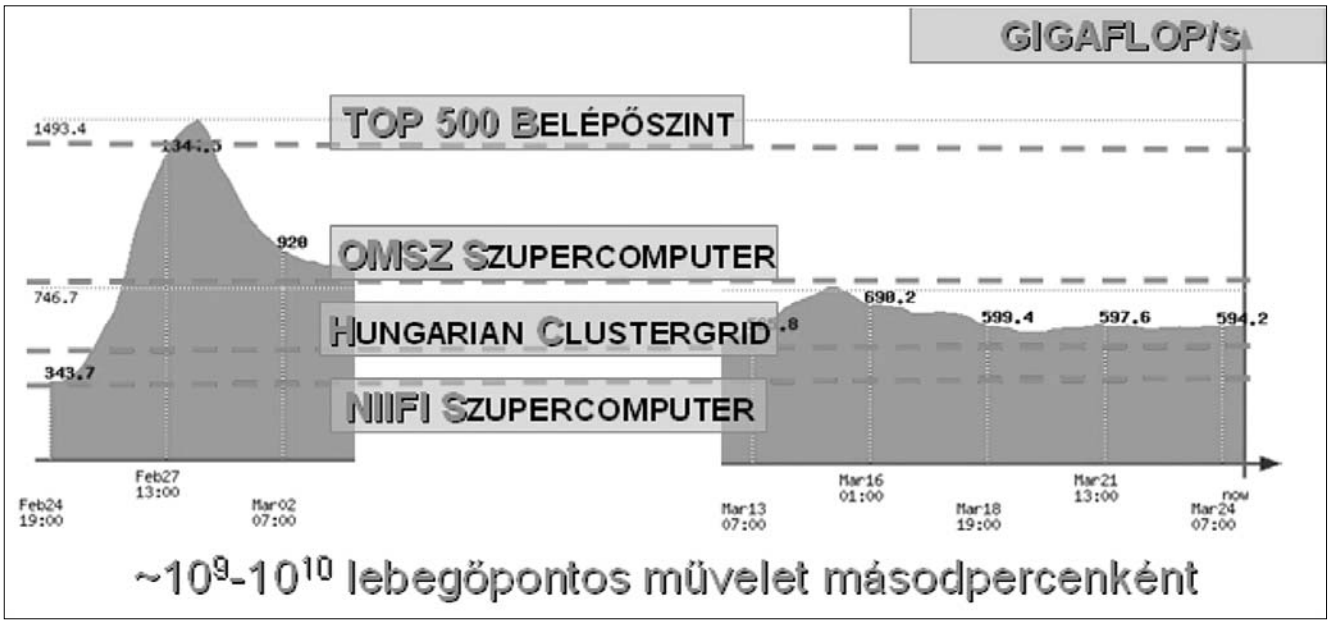
A platformot használó projektek összes teljesítménye 2006. áprilisában mintegy 419 TeraFLOPs [4], mely messze meghaladja a Föld jelenlegi legnagyobb teljesítményű IBM BlueGene/L (280 TeraFLOPs [5]) szuperszámítógép teljesítményét.

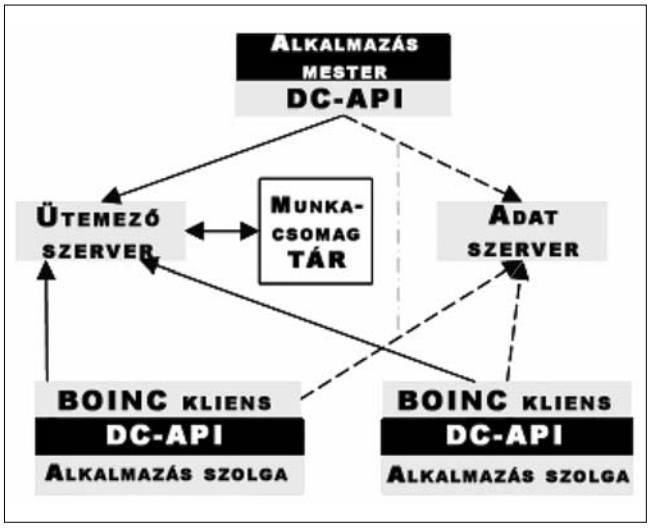
A SZTAKI Desktop Grid [6] globális verziójának felállítása óta eltelt háromnegyed éve az ELTE Komputeralgebra tanszékén futó BinSYS [7,8] projekthez szolgáltat infrastruktúrát. A projekt keretében matematikus kollégáink az összes általánosított bináris számrendszer felkutatását tűzték ki célul. 2005 nyarán, a projekt indulásakor a számrendszereket a 11. dimenzióval bezárólag tűnt reálisnak meghatározni. A 11. dimenzió becsült fél éves feldolgozási ideje a SZTAKI Desktop Grid segítségével 1 hónapra rövidült, továbbá azóta már a 12. dimenzió feldolgozása is megkezdődött. A projektnek jelenleg több mint 7500 résztvevője van, mintegy 18000 számítógéppel. A rendszer által biztosított teljesítmény 700-800 GigaFLOPs *(3. ábra)*, ami majdnem eléri az OMSZ által üzemeltetett leggyorsabb magyar szuperszámítógép teljesítményét, és jelentősen meghaladja az NIIF szuperszámítógépének, valamint cluster gridjének teljesítményét.

A BinSYS projekt szembesített minket azzal, hogy bár a BOINC platform rendelkezik egy programozói API-val, melynek segítségével az alkalmazások felkészíthetőek az elosztott környezetben való futásra, annak használatához elengedhetetlen a BOINC platform átfogó ismerete.

Ez nagymértékben megnehezíti a desktopgrid architektúrára való alkalmazásfejlesztést. Az alkalmazás-

3. ábra SZTAKI Desktop Grid teljesítmény-összehasonlítás





4. ábra Desktop grid infrastruktúra

fejlesztés megkönnyítése érdekében laborunkban kifejlesztettük a DC-API [9] programozói interfészt, melynek feladata elrejteni az alkalmazásfejlesztő elől a BOINC platform sajátosságait. A *4. ábra* a BOINC platformot és a DC-API beépülését szemlélteti.

Az API használata lehetővé teszi, hogy a kutatók saját tudományos kihívásukra koncentráljanak, anélkül hogy a számítási igényeiket kiszolgáló grid infrastruktúrának bármilyen részletével is törődniük kellene.

5. Klaszterek támogatása a SZTAKI Desktop Gridben

A BOINC platform önmagában nem nyújt semmilyen támogatást klaszterek csatlakozására. Mivel a klaszterek 'job management' koncepciója sokkal általánosabb, mint a desktopgridekben alkalmazott részfeladat szétosztás, ezért ez utóbbit át lehet ruházni az előbbire. A BOINC infrastruktúra klaszterekkel való kiterjesztésének öt lehetséges módja van.

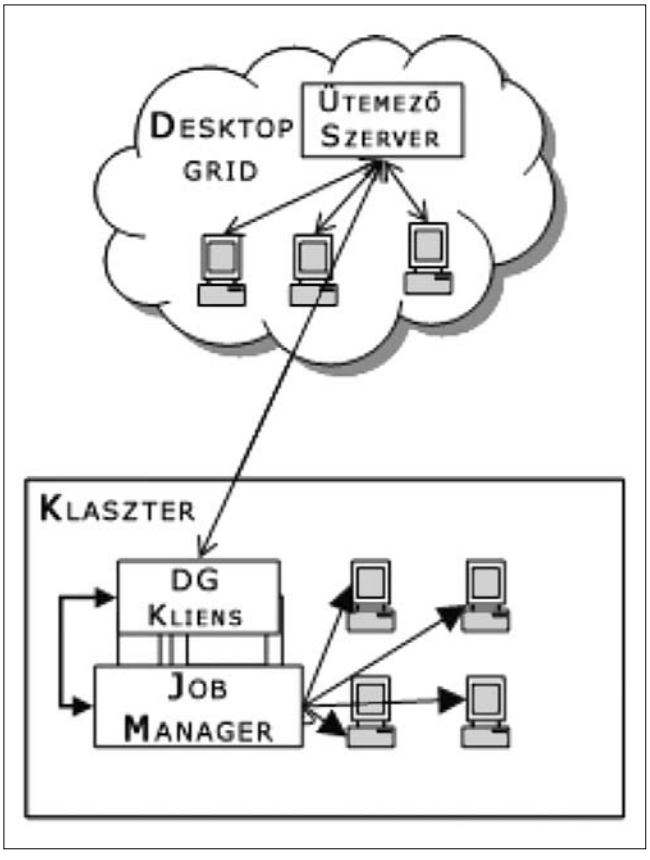
1) A klaszter valamennyi számítógépére telepítjük a BOINC kliens programot, így a klaszter összes erőforrása önállóan csatlakozik a desktopgridhez.
2) Egy teljes desktopgrid telepítésre kerül a klaszterre. A desktopgrid szerver feladatát a klaszter frontend gép látja el. Ez lehetővé teszi a klaszter hierarchikus desktopgridhez való csatlakozását. Erről bővebben a 6. fejezetben szólunk.
3) Egy független magas szintű borker segítségével osztjuk szét a munkát klaszerek és desktopgridek között.
4) A desktopgrid szerver klaszter jelenléte esetén munkacsomagok helyett 'job'-okat küld a klaszternek.
5) A desktopgrid kliens módosított változata kerül telepítésre a klaszter frontend számítógépére, mely a hagyományos desktopgrid munkacsomagokat 'job'-okká alakítja, majd beküldi őket feldolgozásra a klaszterba.

Az első két megoldás hátránya, hogy megkerülik a klaszter 'job manager'-ét, ezért alkalmazásuk nem javasolt. Ennek ellenére, amennyiben a desktopgridek hierarchiába szervezésének lehetősége életre kel, a második megoldás szóba jöhet, mint egy ingyenes lehetőségként klaszterek hierarchikus desktopgridhez való csatolásakor.

A harmadik megoldás hátránya, hogy egy teljesen új brókert kell kifejleszteni hozzá, bár megjegyzendő, hogy egy ilyen bróker segítségével tetszőleges grid architektúra összekapcsolható, amennyiben a különböző koncepciók, reprezentációk és szintaxisok közötti átalakítások megoldhatóak. A negyedik megoldás rendkívül nagy módosítást igényel a desktopgrid szerver oldalán. A szervernek tudnia kell a csatlakozó klaszterről, annak statikus és dinamikus állapot információiról, mely elkerülhetetlenné teszi egy monitorozó rendszer alkalmazását.

Az *5. ábrán* látható ötödik megoldás tűnik a legelegánsabb módszernek klaszterek desktopgridekbe integrálásához. Desktopgrid környezetben a kliensek rácsatlakoznak a szerverre, és munkát kérnek attól. Szakszóval ezt a fajta működést 'pull' üzemmódnak nevezzük. Ezzel szemben a már bemutatott másik architektúrán 'job-manager'-ek küldik be a munkákat a kiválasztott erőforrásoknak, úgynevezett 'push' üzemmódban. Ezzel a megoldással a klaszterek is részt tudnak venni a 'pull' üzemmódú desktopgridek működésében. A desktopgrid klienst módosítva az egyetlen munkacsomag helyett többet is elkérhet egyszerre a szerver-

5. ábra Desktopgridhez csatlakozó klaszter



től, klaszter kompatibilis 'job'-okká alakíthatja azokat, majd beküldheti feldolgozásra a klaszterbe. A desktopgrid szerver egy átlagon felüli teljesítménnyel rendelkező klienst lát csatlakozni. Ehhez a megoldáshoz csak a kliens programot kell módosítani, továbbá mivel az a klaszter 'frontend' gépen fut, ezért az információgyűjtés és a munkabeküldés a klaszterba könnyen megoldható.

Laborunkban ezt a megoldást választottuk, mert ezáltal a klaszterek elegánsan integrálhatóak a desktopgrid architektúrába, megmarad a klaszterek 'job-manager' funkciója, továbbá kevés módosítást igényel a megvalósítása. A SZTAKI Desktop Grid klaszteres verzióját használja a Miskolci Egyetem Általános Informatika Tanszéke a desktopgrid technológia oktatására.

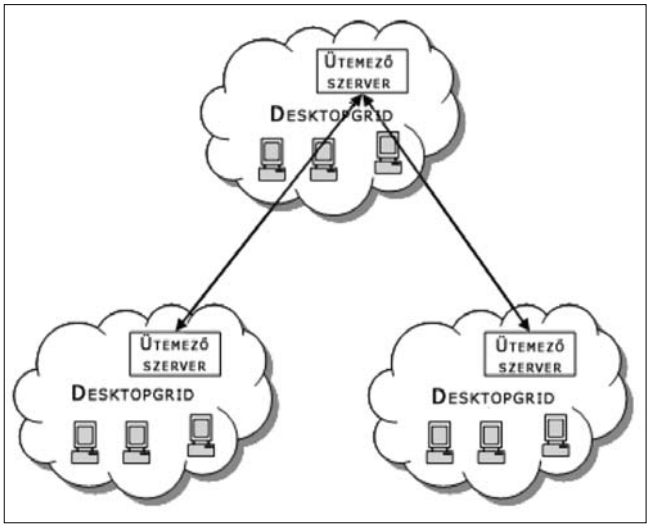
6. Hierarchikus Desktopgrid

Klaszterek támogatásával jelentősen megkönnyítjük a desktopgrid architektúra intézményi elterjedését. Miután sok kisebb intézmény (pl. egyetemi tanszék) önállóan kiépítette saját használatú desktopgridjét, természetes igényként merülhet fel az intézmény egy magasabb szintjén (pl. egyetemi kar) egy nagyobb lélegzetvételű feladat megoldásához a kisebb destkopgrideket összekapcsolva használatba venni. A kiépített desktopgrideket építőelemekként használva lehetőség nyílik azok hierarchiába szervezésére *(6. ábra)*.

Ilyen hierarchiában az alacsonyabb szinten elhelyezkedő desktopgridek munkát igényelhetnek a hierarchiában felettük állótól (pull üzemmód) és fordítva, a magasabb szinten elhelyezkedő desktopgridek munkát adhatnak az alájuk beosztottaknak (push üzemmód).

A SZTAKI Desktop Grid a pull üzemmódot támogatja, mivel ez áll közelebb a desktopgridek működési elvéhez. A magasabb szintről érkező fontos munkáknak prioritáskezeléssel elsőbbség biztosítható, amennyiben a hierarchia ezt megkívánja. A SZTAKI Desktop Grid egy lokális példánya beállítható hierarchikus működésre, vagyis, hogy csatlakozzon rá egy a hierarchiában felette álló SZTAKI Desktop Gridre (a fa struktúrában a szülő elemre). Abban az esetben, ha a hierarchiában a gyermek elemnek kevesebb végrehajtandó munkája akad, mint amennyi szabad erőforrása van, a szülő elemtől további munkát kérhet. A szülő elem egyetlen, nagyteljesítményű kliensként látja a gyermek elemet, analóg módon a klaszterillesztés résznél tárgyaltakhoz. Ehhez természetesen a BOINC szervert olyan funkcionalitással kell kiegészíteni, mely lehetővé teszi annak a kliensként való működést. Ezáltal lehetővé válik az újabb munka kérése, abban az esetben, ha lokálisan nem áll elegendő rendelkezésre.

6. ábra Desktopgridek hierarchiája



A BOINC platform esetében ez könnyűszerrel kivitelezhető. A munkacsomagokat a mester alkalmazás generálja, melyek a BOINC platform saját adatbázisában kerülnek tárolásra. Hogy egy adott munkacsomag egy lokálisan futó alkalmazástól, vagy egy távoli gépről érkezett a BOINC platform számára érdektelen. Elegendő tehát egy új, a szerveren futó démont kifejleszteni, mely figyelemmel kíséri a szerver állapotát. Amennyiben a kliensek munkacsomag kérelme elutasításra kerül, vagyis a szerveren nincs több feldolgozásra váró munkacsomag, a démon BOINC klienst szimulálva további munkacsomagokat kér a hierarchiában felette álló szülő elemtől. Miután megkapta az új munkacsomagokat, nem áll neki a feldolgozásuknak, hanem továbbítja azokat saját szerverének adatbázisába. A démon feladata továbbá az is, hogy figyelje, a hierarchiából kapott munkacsomagok feldolgozásának elkészültét és az eredményeket visszaszolgáltassa a szülő elemnek.

A SZTAKI Desktop Grid hierarchikus verziójának első prototípusa már elkészült, mely egyszintű hierarchiát képes támogatni. Egy most folyó Jedlik Ányos projekt keretében (DTGRID05: Új generációs grid technológiák kifejlesztése és meteorológiai alkalmazása a környezetvédelemben és az épületenergetikában) dolgozunk az általános hierarchikus architektúra megvalósításán.

7. SZTAKI Desktop Grid alkalmazások

Cikkünk végén röviden bemutatjuk azokat a tudományterületeket és alkalmazásokat, melyekhez a SZTAKI Desktop Grid architektúraként szolgál.

A már említett BinSYS projekt mellett nagy számításigény jelentkezik a gyógyszerkutatásban is. Nagymértékű költségcsökkentés érhető el azáltal, ha már a kutatási fázis legelején sikerül kiszűrni a kémiailag instabil, biológiailag inaktív és toxikus molekulákat.

Az ADMEToxGrid [10] projekt keretében a Comgenex cég egy vállalati SZTAKI Desktop Grid felállítását végzi, amit molekulák millióinak ilyen irányú szűrésére fog alkalmazni.

Adatbányászat és mesterséges intelligencia [11] témakörben a Szegedi Egyetemmel karöltve folynak kutatások, egy a SZTAKI Desktop Grid architektúra lehetőségeit kihasználó adatbányász algoritmus kifejleszté-

sében. A projekt innovatív eleme az adatbányász algoritmusok ütemezésének optimalizálása meta-szintű tanulás segítségével. A projekt különös figyelmet szentel az adatvédelemnek. A Szegedi Egyetemen felállított lokális SZTAKI Desktop Griden futó prototípus támogatja az adatbányász projektek helyesség-ellenőrzését, az architektúrának köszönhetően pedig mindemellett bővíthető marad.

A cikk bevezetőjében már említett időjárás-előrejelzés területen a lokális SZTAKI Desktop Grid architektúrát szolgáltat a Magyar Meteorológiai Szolgálatnak numerikus időjárás-előrejelzési és klímamodellezési alkalmazásokhoz [12].

A SZTAKI Desktop Grid projekt nemzetközi sikerét követően, második nemzetközi eredményünk az Egyesült Királyság két egyetemével – a Westminsteri Egyetemmel és a Brunneli Egyetemmel – való együttműködés [13]. Az együttműködés keretében digitális jelfeldolgozás témakörben folyik kutatás periodikus, nem egyenletes eloszlású mintavételi szekvenciák tervezésének területén a digitális jelfeldolgozás zavarmentessé tételéért. A két egyetemen felállításra került egy 100 személyi számítógépből álló desktopgrid, mellyel az addig egyetlen számítógépen futó algoritmus 1 hónapos feldolgozási idejét sikerült 2 napra csökkenteni.

A kutatások mellett a SZTAKI Desktop Grid architektúra a miskolci egyetemen oktatás tárgyát is képezi a Párhuzamos és Elosztott Rendszerek tárgy keretein belül, melynek során a tárgy hallgatói megismerkednek a desktopgrid architektúrára való alkalmazásfejlesztés rejtelmeivel. Az egyetemen felállításra került egy lokális SZTAKI Desktop Grid rendszer, mely a diákok tanulmányi munkáján kívül az egyetemen folyó kutatómunkát is támogatja.

8. Összefoglalás

A cikkben bemutatásra került a BOINC platformra épülő SZTAKI Desktop Grid. A grid személyi számítógépeket építőelemként felhasználva lehetővé teszi számításigényes feladatok megoldását. Külön került bemutatásra a klaszterek támogatásának lehetősége, egy módosított kliens program segítségével, mely a BOINC platform által generált munkacsomagokat hagyományos ‘job’-bá konvertálva átadja azokat a klaszter ‘job manager’-énak. Az építkezést tovább folytatva tárgyaltuk továbbá a desktopgridek hierarchiába szervezésének lehetőségét.

A lehetőség, hogy desktopgridek egymás között átadjanak munkacsomagokat egymásnak egy újabb lépés egy olyan grid infrastruktúra kialakításában, mely könnyen telepíthető és melyhez bárki felajánlhatja erőforrását; két olyan tulajdonság mely napjaink grid rendszereiben egyszerre nem lelhető fel. A SZTAKI Desktop Grid technológiát mind az oktatási, mind a kutatási intézmények használhatják saját gridük felállítására, de mint a Comgenex példája is mutatja, cégek számara is hasznos vállalati gridek létrehozására.

Az adott intézmény már rendelkezésre álló gépeinek bevonásával, minimális költséggel – egy kis szerver üzembeállításával – szuperszámítógép teljesítmény érhető el. A fentiek alapján javasoljuk az oktatási és kutatási intézményeknek, illetve vállalatoknak saját Grid felállítását, amiben a SZTAKI kész a szükséges segítség megadására.

Köszönetnyilvánítás

A szerző köszönetét fejezni ki Kacsuk Péternek, az MTA-SZTAKI Párhuzamos és Elosztott Rendszerek Kutatólaboratórium vezetőjének, Kovács Józsefnek a SZTAKI Desktop Grid projekt vezetőjének, továbbá munkatársainak, Gombás Gábornak, Marosi Attilának és Vida Gábornak a cikk megírásához nyújtott sok segítségért és szakmai támogatásukért.

Irodalom

[1] EGEE: Enabling Grids for E-SciencE. http://public.eu-egee.org/
[2] D.P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, D. Werthimer: SETI@home: An Experiment in Public-Resource Computing, Communications of the ACM, Vol. 45 No. 11, November 2002, pp. 56–61.
[3] D. P. Anderson: BOINC: A System for Public-Resource Computing and Storage. 5th IEEE/ACM International Workshop on Grid Computing, November 8, 2004, Pittsburgh, USA. Available at: http://boinc.berkeley.edu/grid_paper_04.pdf
[4] BOINCstats.com. http://boincstats.com
[5] TOP500 Supercomputer Sites. http://www.top500.org
[6] SZTAKI Desktop Grid. http://szdg.lpds.sztaki.hu/szdg, http://www.desktopgrid.hu
[7] A. Kovács PhD, P. Burcsi, J. Kasza, N. Podhorszki PhD, G. Vida, A. Kornafeld, Generalized binary number systems. http://compalg.inf.elte.hu/projects/binsys/
[8] Kornafeld, A., Kovacs, A.: Szuperszámítógépes teljesítmény szuperszámítógép nélkül – A BinSYS projekt –, Networkshop 2006, Miskolc, NIIF, 2006. Elérhető: http://www.lpds.sztaki.hu/~kadam/BinSYS.pdf
[9] Podhorszki, N., Vida, G.: Alkalmazói programozási felület SETI-jellegű elosztott programokhoz és végrehajtó rendszer a BOINC infrastruktúrára, Networkshop 2005, Szeged. NIIF, 2005. Elérhető: http://www.lpds.sztaki.hu/~pnorbert/pub/desktopgrid-nws05.pdf
[10] ADMEToxGrid. http://www.admetoxgrid.hu
[11] További információ: http://www.sztaki.hu/search/projects/project_information/?uid=00025
[12] További információ: http://www.sztaki.hu/search/projects/project_information/?uid=00188
[13] Gridalliance. http://www.gridalliance.org.uk