Multidiszciplináris tudományok, 4. kötet. (2014) 1. sz. pp. 147-158.

HÁLÓZATI FORGALOMKEZELÉS SZÁMÍTÁSI FELHŐ SZIMULÁTOROKBAN

Kecskeméti Gábor
Posztdoktorális kutató, Innsbrucki Egyetem, Informatikai intézet,
Elosztott és Párhuzamos rendszerek kutatócsoport
Cím: Ausztria, 6020, Innsbruck, Techkikerstrasse 21a, email: gabor@dps.uibk.ac.at
Egyetemi tanársegéd, Miskolci Egyetem, Általános Informatikai Tanszék
Cím: 3515 Miskolc, Miskolc-Egyetemváros, email: kecskemeti@iit.uni-miskolc.hu

Összefoglalás
Az infrastruktúra felhő szolgáltatások felhasználói igény szerinti virtuális infrastruktúrákat érhetnek el akár minimális informatikai infrastruktúra fenntartási gyakorlatta isl. A szolgáltatások megbízhatósága és rugalmassága olyan szintű központosításhoz vezet amely gátolja az azok működését és fejlesztését célzó kutatásokat közvetlenül ezeken az éles rendszereken. Ezért a kutatók szimulátorok segítségével ellenőrzik a jövőbeli felhő technológiákat célzó újításaikat. A szimulátorok pontossága döntően befolyásolja az újítások alkalmazhatóságát. Ez a cikk rámutat arra, hogy a hálózatkezelés terén a felhő szimulátorok sokszor nagy eltéréseket mutatnak a valóságtól. Ezen eltérések vizsgálatából eredeztetve pedig bemutat egy olyan új szimulációs eljárást, amely nemcsak növeli a szimulációs pontosságot hanem még teljesítménye sem marad el a korábbi megoldásoktól, így továbbra is általánosan alkalmazható.
Kulcsszavak: *számítási felhő, szimuláció, hálózatkezelés*

Abstract
Infrastructure as a service systems offer access to on demand virtual infrastructures even for people with minimal IT maintenance experience. The reliability and flexibility of these systems lead to high level of centralization that limits researchers to improve the internal behaviour of such systems. Thus, researchers evaluate their novel improvements with simulators. The precision of these simulators heavily influence the real-life applicability of the new findings. This article shows that current cloud simulators often have significant errors when comparing their network behaviour to real life. Based on the analysis of these errors, this article introduces a new simulation method that not only increases the precision of the simulation but also maintains the performance of past solutions (allowing its general use).
Keywords: *cloud computing, simulation, network management*

1. Bevezetés

Az infrastruktúra szolgáltatások a számítási felhők egyik első fontos alága. Olyan alapvető szolgáltatásokat biztosítanak, amelyek lehetővé teszik felhasználóik számára egy virtuális számítási infrastruktúra létrehozását, kezelését, megszüntetését. Mindezen műveletek automatizált módon végezhetőek. Az infrastruktúra szolgáltatások alapvetően az erőforrás virtualizációs technológiák és web szolgáltatások ötvözéseként jöttek létre. Az erőforrás virtualizáció egy a 70-es évek óta aktívan fejlesztett technológia mely a 90-es évek közepé-

től kezdett el szélesebb körben elterjedni az Intel 32 bites architektúrájával kompatibilis processzorokkal ellátott rendszereken. A korai megoldások hatékonyságbeli problémái megakadályozták a vállalati használatot, de a paravirtualizáció (Xen) majd pedig a hardverrel támogatott virtualizáció segítségével egyre több vállalat kezdte ezeket a megoldásokat alkalmazni. Először virtuális privát szerverparkok jöttek létre amelyek segítségével a tetszőleges informatikai infrastruktúra bérlését tették lehetővé a cégek. Ezen megoldásokat még főleg kézzel vezérelték a virtualizációs megoldások rendszeradminisztrátorai. Később megjelentek az első olyan megoldások, amelyek segítségével web szolgáltatás felületről automatikusan lehetett a bérlendő infrastruktúra paramétereit (hálózati sávszélességek, topológia – VLAN –; processzor, memória és háttértár követelmények virtuális gépenként) beállítani és a kívánt időpontban használatba venni. Az akadémiai megoldások közül elsőként a Virtual Workspace Service (későbbiekben Nimbus), míg ipari környezetben elsőként az Amazon (EC2 néven) nyújtott ilyen szolgáltatást.

A korai ipari alkalmazásnak köszönhetően az infrastruktúra felhők hamar nagy népszerűségre tettek szert. Mivel szerződésben garantált szolgáltatásokat nyújtanak rajtuk így a kutatási jellegű kísérleteknek ezeken a rendszereken helye nincs. A kutatás támogatásához ugyanis virtuális infrastruktúrákat nyújtó fizikai eszközök újabb és újabb kísérleti eszközökkel és szoftverekkel történő ellátásara lenne szükség (például új virtuális gép ütemező, vagy energia fogyasztás mérők rendszerszintű beiktatása). Az ilyen, az infrastruktúra szolgáltatások megreformálására törekvő kutatásokat ezért vagy a vállalati felhők többszázezer processzoros rendszereinél nagyságrendnyivel kisebb egyedi módon konfigurálható fizikai infrastruktúrákon végzik, vagy pedig olyan felhő szimulátorokat használnak, amelyek képesek elfogadható teljesítménnyel (pl. néhány fizikai gépen) a vállalati felhők erőforrásaihoz mérhető rendszerek egyes paramétereit megfelelő pontossággal meghatározni.

Azon kutatások eredményei, amelyek a szimulációkra alapozva születnek viszont sok esetben a használt szimulátor pontosságától és részletességétől függnek. Az infrastruktúra felhő szimulátorok (mint pl. a CloudSim [9] vagy a GroudSim [8]) többnyire korábbi szimulátorokra építenek és mindösszesen kiegészítik azokat az infrastruktúra felhők főbb jellegzetességeit magában foglaló szimulációs modulokkal. A jelenleg elérhető szimulátorokat alapvetően két csoportra bonthatjuk: *(i)* hálózati szimulációt kiegészítő, *(ii)* feladatütemezési szimulációt kiegészítő.

A hálózati szimulációkra alapuló (pl. NS2-3 [10], OMNET++ [1]) szimulátorok nagy részletességgel tartalmazzák a hálózat működését ami miatt a szimulációk létrehozása nehézkes (akár több tízezer hálózati csomópont pontos hálózati viselkedését kellene meghatározni a szimulátor számára). Viszont az így létrehozott szimulációk pontosságához nem férhet kétség (habár az időigényük rendkívül nagy). Sajnos a kutatók többnyire elnagyolják a hálózati modell pontos megadását és így sokszor hiába építettek a hálózati szimulátorokra az eredmények megbízhatatlanok lesznek.

A feladatütemezési szimulációkat (pl. GridSim [7], SimGrid [4]) kiegészítő szimulátorok ezzel szemben többnyire igen elnagyolt hálózati modellt tartalmaznak. A jelenlegi felhő szimulátorok a szimulációs idő röviden tartásának érdekében az elnagyolt modellek kiterjesztésére nem is törekedtek. Tehát amennyiben valamilyen hálózati forgalom fellép a szimuláció során azt vagy a szimulátor felhasználójának kell lemodelleznie, vagy pedig egy másik szimulátort kell választania.

Annak érdekében, hogy megértsük, milyen szituációkban lenne fontos a hálózat pontosabb modellezése, ez a cikk először bemutatja, hogy a számítási felhőkben milyen egymás-

ra hatása lehet a virtuális infrastruktúra létrehozásának más hálózati műveletekre. Első lépésként a cikk bemutat néhány releváns virtuális infrastruktúra létrehozási eljárást és annak hatását a hálózati forgalomra. Második lépésként a cikk bemutatja, hogy a jelenlegi széles körben használt szimulátorokkal ezek a hatások nem figyelhetőek meg a szimulált hálózati forgalom alakulásában.

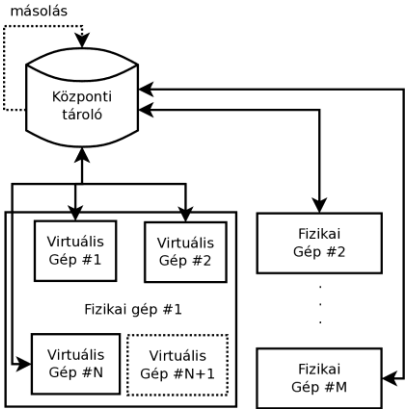
A cikk ezután felvázol egy olyan szimulációs megoldást (amit a későbbiekben DISSECT-CF-nek hívok majd), amely a használhatóságot nem rontja szignifikánsan (teljesítménye a feladatütemezési szimulátorokra alapuló megoldásokkal összemérhető), de mégis képes a virtuális infrastruktúrák létrehozása alatti megváltozott hálózati viselkedést pontosabban meghatározni. A javasolt megoldás első lépésben kiterjeszti a virtuális gépek létrehozási modelljét a virtuális gépek első életszakaszának helyesebb szimulációjával (a virtuális gép lemezképeinek kezelése bekerült a szimulálandó feladatok közé). Második lépésben a lemezkép kezelés hatásainak vizsgálatát lehetővé teendő, a DISSECT-CF a felhasználói viselkedés hálózati aktivitásának szimulációját együtt végzi a lemezkép kezelésből adódó hálózati forgalommal. Ezáltal lehetővé válik megfigyelni, hogy mások virtuális infrastruktúra létrehozási műveletei milyen befolyással vannak egy-egy a hálózatot is aktívan használó feladatnak a végrehajtására.

2. Problémafelvetés és irodalomkutatás

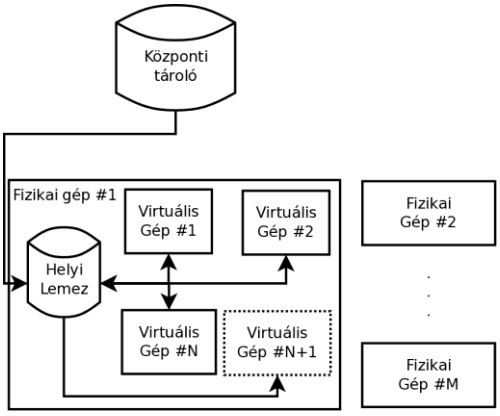
Alapvetően két fő esetet különbözethetünk meg, amikor az infrastruktúra felhők hálózati szolgáltatásait igénybe veszi egy felhasználó: *(i)* a virtuális infrastruktúrája létrejöttekor, *(ii)* a virtuális infrastruktúra használata során. A jelenlegi felhő szimulátorok a kizárólag a második esetre koncentrálnak (és ilyen forgalom generálására adnak lehetőséget a felhasználóik számára). Azonban azokban az esetekben, amikor virtuális infrastruktúra létrehozás és használat párhuzamosan történik, a jelenlegi megközelítés nem nyújt megfelelő pontosságot. Az ok egyszerű: az létrehozási műveletek olyan hálózati forgalmat generálhatnak, amely szignifikánsan befolyásolhatja a hálózat teljesítménymutatóit a használat során. Ezért egy helyes hálózati működésű szimulátornak mindenképpen figyelembe kell venni a létrehozási műveletek hatásait a hálózatra. Így adódik a kérdés milyen esetekben hatnak egymásra a létrehozási és használati műveletek, és hogyan lehetséges ezeknek a hatékony beépítése egy szimulátorba.

2.1. Az egymásrahatás vizsgálata

Az egymásrahatás megértéséhez, első lépésként megvizsgáljuk mikor és hogyan használják a hálózatot a létrehozási műveletek. A virtuális infrastruktúra létrehozása a következő fontosabb lépésekből áll: *(i)* virtuális gép helykeresés, *(ii)* rendszerkép előkészítés, *(iii)* VLAN konfigurálás. Ezek a lépések mind bizonyos szintű hálózati forgalommal járnak, de a rendszerkép előkészítési lépéshez képest a többi művelet elhanyagolható forgalmat jelent. Tehát, ha valami befolyásolhatja az infrastruktúra használat hálózati műveleteit az csak a rendszerkép előkészítés lehet. A rendszerkép előkészítésnek két fő módozata van: *(i)* új virtuális gép létrehozása és *(ii)* korábban használt virtuális gép újrahasznosítása.



Ábra 1 Központzi lemezkép másolat használat



Ábra 2 Helyi lemezkép másolat használat

2.1.1. Virtuális gépek létrehozásának hálózati terhelése

A virtuális gépek létrehozatalakor az infrastruktúra felhő szolgáltatások mindig készítenek egy másolatot egy központilag tárolt virtuális egyenképből (VA). Attól függően, hogy a felhőszolgáltató hol tárolja a másolatokat, a hálózatot két különböző terhelés éri.

Ha a másolatok a virtuális gépet futtató fizikai gépre kerülnek, akkor a központi tárból a fizikai gépre másolás a virtuális gép indulása előtt fog nagyobb adatforgalmat generálni a hálózaton (lásd: Ábra 2). A mai VA-k mérete tipikusan néhány gigabájtban mérhető. Ennek az adatmennyiségnek kell a hálózaton átmozognia minden egyes virtuális gép indulása előtt, ha a fizikai gépeken történik a másolatok tárolása.

Ha a másolatokat a központi tárolón készítik, akkor ez nem generál közvetlenül hálózati forgalmat (lásd: Ábra 1). Viszont a következő következményekkel jár: minden egyes virtuális gép a futása során, amikor lemezműveleteket végez automatikusan a központi tárolóhoz fog fordulni. Tehát a virtuális gépek futtatása folyamatos hálózati forgalmat generál (a fizikai gépre került gépek csak akkor használták a hálózatot amikor a felhasználónak szüksége volt rá). Ráadásul, amikor egy virtuális gép elindul akkor a központi tárolón komoly ideiglenes terhelés keletkezik (hiszen duplikálni kell a virtuális géphez tartozó VA-t). Ez alatt az ideiglenes terhelés alatt a felhőszolgáltatás által korábban befogadott és futó virtuális gépek lemezműveletei szenvednek el lassulást. A lassulás oka, hogy osztozniuk kell a a központi tároló lemez alrendszerének sávszélességén az éppen nagymennyiségű adatot duplikáló rendszerkép előkészítési folyamattal. Tehát az ideiglenes terhelés alatt a lemezműveletet végző virtuális gépek lassulására lehet számítani központi tárolás esetén. A hálózati forgalomra ez a hatás olyan befolyással van, hogy a nem lemezműveleteket végző virtuális gépek hálózati hozzáférése ideiglenesen javul.

2.1.2. Virtuális gépek újrahasznosításakori hálózati terhelés

A virtuális gépek újrahasznosíthatóságának előfeltétele, hogy egy korábbi virtuális gép leállításakor az infrastruktúra felhő szolgáltatás fel legyen szólítva a virtuális gépnek korábban létrehozott (és azóta a virtuális gép által megváltoztatott) lemezkép másolat törlésének elhagyására. Ha ez a feltétel teljesült akkor a virtuális gép újrahasznosítható mert az infrastuktúra felhő megőrzi a korábbi lemezállapotát. A lemezkép másolat tárolási helyétől függően az újrahasznosítás is más-más módon hathat a hálózat teljesítménymutatóira.

Ha a másolat a korábbi virtuális gépet futtató fizikai gépen található, akkor az infrastruktúra felhőknek el kell döntenie, hogy futtatható-e az újrahasznosított virtuális gép a korábbi fizikai gépen vagy sem. Amennyiben futtatható, akkor nem kell számolni különösebb hálózati forgalommal (tehát egy ilyen jellegú újrahasznosítást használó és újrahasznosítás domináns szimulációban a korábbi szimulátorok eredményei is érdemlegesen használhatóak). Azonban, ha az újrahasznosításra egy másik fizikai gépen kerül sor, akkor először az infrastruktúra szolgáltatásnak át kell mozgatnia a korábbi másolatot az újonnan kiválasztott fizikai gépre. Ez az átmozgatás hasonló hálózati forgalmat eredményez, mintha egy teljesen új virtuális gép lenne létrehozva az újonnan kiválasztott fizikai gépen. Az egyetlen különbség, hogy a forgalom nem a központi VA tár és az újonnan kiválasztott fizikai gép között emelkedik meg, hanem a korábbi fizikai gép és az újonnan kiválasztott között.

Ha a másolat a központi tárolón található, akkor a virtuális gép újrahasznosítás folyamata nagyban leegyszerűsödik: a gép azonnal indulhat, hiszen a másolata már ott található a központi tárban ahogy azt várta. Az így újrahasznosított virtuális gépek ezután teljesen hasonló hálózati viselkedéssel rendelkeznek, mint egy újonnan létrehozott virtuális gép amelynek a központi tárolón található a lemezképe.

2.2. Jelenlegi szimulátorok és az egymásrahatás kimutathatósága

Először az egyik legszéleskörűbben használt szimulátort a CloudSim-et [9] vizsgáltam meg. Ez a szimulátor a szerzők által korábban készített még a Grides számítások világának követelményeihez igazított szimulátorra a GridSim-re [7] alapszik. A Grides környezethez hozzáadták az virtualizált adatközpontok lehetőségét amelyekre építve nagyméretű elosztott infrastruktúra felhő rendszer hozható létre a szimulációkban. A modellezés főképp a számítási feladatok elvégzésere és a különböző adatközpontok közötti információcserére koncentrál (ezt az információcserét is feladatok előtt vagy után lehet csak közvetlenül elvégeztetni vele). A NetworkCloudSim [3] nevű kiterjesztéssel pótolták azt a hiányosságot, hogy adatközpontokon belül semmilyen módon nem tudott adatmozgásokat modellezni a szimulátor. Sajnálatos ez a kiterjesztés továbbra is a számítási feladatok kapcsán végzi az adatközpontokon belüli hálózati modellezést. Így a feladatokhoz közvetlenül nem kötődő virtuális infrastruktúra létrehozási műveletek nem szimulálhatóak. Ezáltal sem a CloudSim-mel sem pedig a kiterjesztésének segítségével nem lehet olyan szimulációt készíteni, amely a számítási feladatok kommunikációjával párhuzamos a virtuális infrastruktúra létrehozási feladatok hálózati teljesítményre gyakorolt hatását helyesen modellezi.

A GreenCloud [6] az egyik legújabban elérhető szimulátor. Így jobban a mai trendekhez igazodik és nekik megfelelően precíz energia és hálózati modellezést igér. A népszerű NS2 hálózati szimulátort terjeszti ki felhőkben alkalmazott adatközpont szervezési eljárásokkal. Így megkönnyíti a felhőkben a precíz hálózati szimulációkat. Mivel azonban hálózati szimulátorra alapoz, érdemleges szimulációhoz az alkalmazások igen pontos leírására van szüksége a használójának. Ez erősen leszűkíti az alkalmazhatóságát hiszen olyan esetekben amikre a használó nem gondolt, a szimulátor nem képes semmilyen eredménnyel szolgálni (pl. ha a szimuláció egy bizonyos fájlátviteli protokollt implementál, más protokollokra nem adaptálható könnyen az eredmény). Mivel a virtuális infrastruktúra létrehozás során a különféle infrastruktúra felhők külön utakat járnak, a szimulátor virtuális gép modelje csak a minimális részletekre koncentrál ami például a virtuális gép létrehozása során keletkező hálózati forgalom szimulációjára nem terjed ki.

A GroudSim [8] egy Innsbrucki Egyetem által fejlesztett grid és felhő környezeteket egyaránt modellezni képes szimulátor. Egyik legfontosabb fejlesztési szempontja az volt, hogy a korábbi szimulátorok sebességbeli problémáit orvosolja. A hálózati és a számítási műveletek szimulációját egységesen végzi mind grid mind infrastruktúra felhők esetén (virtuális gridet hoz létre a felhőn). A hálózati modellje egyszintű topológiát kezel (minden gép és virtuális gép közvetlen kapcsolattal rendelkezik a többivel), és a hálózati kapcsolatok kizárólag fájlműveletekre használhatók. A kapcsolatok igazságos elosztása nem történik meg: azaz minden fileátvitel a kapcsolat rá eső hányadát használhatja csak (ha nem használja ki a rá eső hányadot akkor a sávszélesség veszendőbe megy a szimuláció szerint). A virtuális gépek létrehozása és megszüntetése nem generál hálózati forgalmat, és ez a jelenlegi hálózati modell egyszerűsége ellenére nem is lenne megvalósítható koncepcionális változtatások nélkül.

Következőként az iCanCloud [2] nevű szimulátort tekintjük át. Ez egy OMNET++ [1] keretrendszerre épülő megoldás ami a felhő felhasználókat veszi célba. Ez nagyban megkülönbözteti a többi szimulátortól amelyek főleg a számítási felhőkben felmerülő problémák kutatását támogatják. Az iCanCloud segítségével a felhasználók eldönthetik milyen időpont a legalkalmasabb egy-egy felhőben végzendő feladat ellátására mind pénzügyi mind teljesítménybeli megfontolások alapján. Mivel ez a szimulátor felhasználó központú ezért a felhők belső működése csak a számukra szükséges mértékben kidolgozott a maximális teljesítmény érdekében.

Végül a SimGrid [4] egy tradícionális grid szimulátor. A számítási felhők előretörésével ennek is készült virtualizációt magába foglaló kiterjesztése. Ez a kiterjesztés hasonlóan a GroudSim-hez és a CloudSim-hez a grid a felhőben koncepcióra épít és a szimulációt a grides környezetekre vezeti vissza. Habár a hálózatkezelésére különös figyelmet fordítottak [5], a grid a felhőben koncepció miatt, a virtuális infrastruktúra létrehozásának hálózati hatásai nem jelennek meg ebben a szimulátorban sem megfelelő részletességgel. Ezen felül a felhőre koncentráló kiterjesztés évek alatt sem nyerte el végső formáját, így ilyen környezetbeli használata nem javasolt.

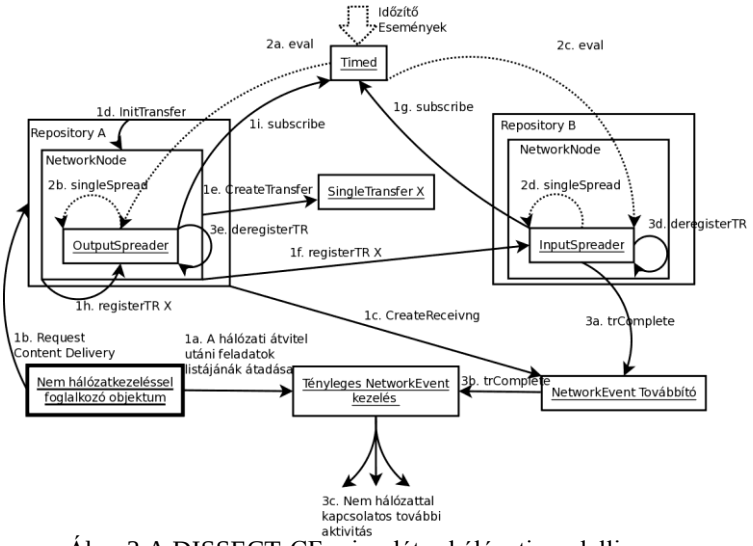
A felhőszimulátorok kevésbé elterjedt ága az adatközpontok működtetési problémáira koncentrál [11-13]. Ezek a szimulátorok mind részletekbe menő szimulációkat tudnak véghezvinni amik lehetővé teszik például IaaS megoldások skálázhatósági problémáinak vizsgálatát vagy az adatközpontok hűtő rendszereinek méretezési problémáinak elemzését is. Ezen szimulátorok azonban jelentősen eltérnek a tárgyalt tématerülettől ezért a későbbiekben nem foglalkozom velük.

3. Egy újfajta szimulációs megközelítés

A DISSECT-CF egy olyan eseményvezérelt szimulátor koncepció amely kifejezetten az infrastruktúra felhők belső működésének tanulmányozásának lehetővé tételét tűzte ki céljául (a cikk a szimulátor 0.7-es verziójának működését tekinti át; a szimulátor aktuális verziója a következő linken érhető el: *https://github.com/kecskemeti/dissect-cf*). Segítségével olyan kutatások valósíthatóak majd meg amelyekben az infrastruktúra felhők belső átszervezésének hatásai vizsgálhatók azok felhasználóira. A következőben álljon itt néhány példa a kitűzött szimulációs feladatokból: *(i)* hogyan befolyásolja egy másfajta virtuális gép elhelyező algoritmus használata a felhasználói elégedettséget, *(ii)* milyen virtuális egyengép (VA) másolás optimalizáció eredményezi a legkisebb hálózati terhelést az infrastruktúrá-

ban, *(iii)* van-e lehetőség magasszintű virtuális gép ütemezéssel hatékonyabb energiafelhasználású virtuális infrastruktúrát fenntartani ahoz képest, hogy az ütemezést teljesen az infrastruktúra felhőre bízzuk stb. Jónéhány felmerülő kérdés esetében elengedhetetlen a helyes hálózati szimuláció (amely magában foglalja az infrastruktúra felhő működéséből adódó virtuális infrastruktúra létrehozás kapcsán felmerülő hálózati terhelést is). Ezért a DISSECT-CF egy újfajta hálózat szimulációs megoldást tartalmaz, amely hasonlóan a nem hálózati szimulátorokra alapozó felhő szimulátorokhoz, folyamszintű hálózati forgalom szimulációt tesz lehetővé. A folyamszintű szimulációban a hálózat csomag alapú működése nem kerül kivitelezésre, hanem csak a hálózati kapcsolatok áteresztőképessége és a TCP/IP protokoll igazságos sávszélesség elosztásra törekvése alapján történik a hálózati kommunikáció átviteli képességeinek szimulálása.

3.1. A sávszélesség igazságos elosztása



Ábra 3 A DISSECT-CF szimulátor hálózati modellje

Az Ábra 3-on a DISSECT-CF hálózati modellje látható. A hálózati működést alapvetően 5 komponens határozza meg: *(i)* a `Repository`, *(ii)* a `NetworkNode`, *(iii)* az `OutputSpreader`, *(iv)* az `InputSpreader` és *(v)* a `SingleTransfer`. A `Repository` valósítja meg a központi tárolók és a fizikai gépek helyi lemezek kapacitás kezelését (regisztrálhatóak beléjük adatok, és lekérhetőek onnan), ezen felül rajtük keresztül lehet fájlátviteleket kivitelezni a rendszerben. A DISSECT-CF szimulációkban hálózati forgalom csak `NetworkNode`-ok között lehetséges. A `Repository` a `NetworkNode` osztály kiterjesztése, így önnállóan képes kommunikálni a külvilággal. A fizikai és virtuális gépek ezen `Repository`-k magukban foglalásával (a helyi lemezük reprezentálásaként) válnak képessé a kommunikációra. A `NetworkNode` osztály tetszőleges adatkommunikációt tesz lehetővé (nincs korlátozva fájlátvitelre). Minden kommunikáció alapja a `SingleTransfer` osztály, amelynek példányai reprezentálnak egy-egy adatátvitelt a szimulált rendszerben. Az `OutputSpreader` és az `InputSpreader` osztályok a ki- és bemeneti kapcsolatait

reprezentálják egy csomópontnak. Ezek az osztályok felelősek a hálózat igazságos kihasználásáért. Mindkét osztálynak a fő tulajdonsága az áteresztőképessége (byte/sec-ben kifejezve). A ki- és bemeneti sávszélesség igazságos elosztása a következő lépésekben történik:

1. Feltételezük, hogy minden `SingleTransfer` ugyanakkora sávszélességet kap.
2. A `singleSpread` függvény meghívásával elosztjuk a jelenlegi adatátviteleket reprezentáló `SingleTransfer` objektumok között az aktuális időszelet sávszélességét.
3. Ha valamely `SingleTransfer` objektum nem tudja az rá eső teljes sávszélességet felhasználni, akkor az előző 2 lépést újra megismételjük a fel nem használt sávszélességgel.

Hálózati hierarchia úgy hozható létre, hogy olyan `NetworkNode` objektumokat készítünk emelyek a `SingleTransfer` objektumokon jelzik a számukra elérhető maximális sávszélességet. Azaz ha egy `SingleTransfer` objektum nem közvetlenül összekapcsolt csomópontok közötti kommunikációért felel, akkor a köztes `NetworkNode` objektumok nem hagyományos `singleSpread` műveleteket végeznek hanem olyanokat amik csak jelzik a a forrás és a célcsomópontnak, hogy maximum mekkora sávszélességet használhatnak fel az adott adatátvitelre.

Alapértelmezetten minden szimulált adatközpontban kétrétegű hierarchiát épít fel a DISSECT-CF. Adatközpontonként létrehoz egy vagy több központi tárat (ezekbe történik az adatközpontban elérhető virtuális egyengépek (VA) regisztrációja). Ezeket a központi tárakat és a fizikai gépeket (amik a virtuális gépeket fogják a későbbiekben futtatni) egy közös hálózatba szervezi. Ez a hálózat rendelkezik egy külső internet kapcsolattal. A hálózat második rétege a fizikai gépek és az általuk futtatott virtuális gépek között jön létre. Az elsőszintű halózathoz az átjárót a fizikai gépek jelentik a virtuális gépek számára. Ezzel a hálózati hierarchiával garantált, hogy virtuális gépek közötti kommunikáció és a virtuális infrastruktúra létrehozási műveletei egymásra tudnak hatni. Például, ha egy virtuális gép adatot forgalmaz egy másik fizikai gépen található virtuális géppel, akkor az első szintű hálózat forgalmi viszonyait megváltoztató virtuális infrastruktúra létrehozási művelet kihatással lehet a két virtuális gép közötti kapcsolat teljesítménymutatóira.

3.2. Egy hálózati átvitel lépésenkénti bemutatása

Az Ábra 3-on egy DISSECT-CF által modellezett hálózati átvitel modellezési lépései is láthatóak. Ezek a lépések három fázisra bonthatóak: *(i)* átvitel előkészítés, *(ii)* átvitel, *(iii)* átvitel befejezés. Az ábra minden egyes szimulációs lépést egy nyíllal jelöl. A nyíl felirata mindig egy szám és egy betű kombinációjával kezdődik, a szám a fázist azonosítja, a betű pedig a fázison belüli szekvenciát. A folyamatos nyilak egyszer végrehajtott lépéseket jelentenek, míg a szaggatott nyilak olyan lépéseket jelentenek, amelyek tetszőleges alkalommal megismételhetőek.

3.2.1. Átvitel előkészítés

Az átvitel előkészítési fázisban láthatjuk a tényleges felhasználóját a hálózati átvitelnek: „Nem hálózatkezeléssel foglalkozó objektum” néven. Amikor ez az objektum eldönti, hogy fájlátvitelre van szükség akkor első lépésként (*1a.*) létrehoz egy olyan eseménykezelőt amibe elhelyezi azokat a feladatokat amiket a hálózati átvitel befejezésekor szeretne elvé-

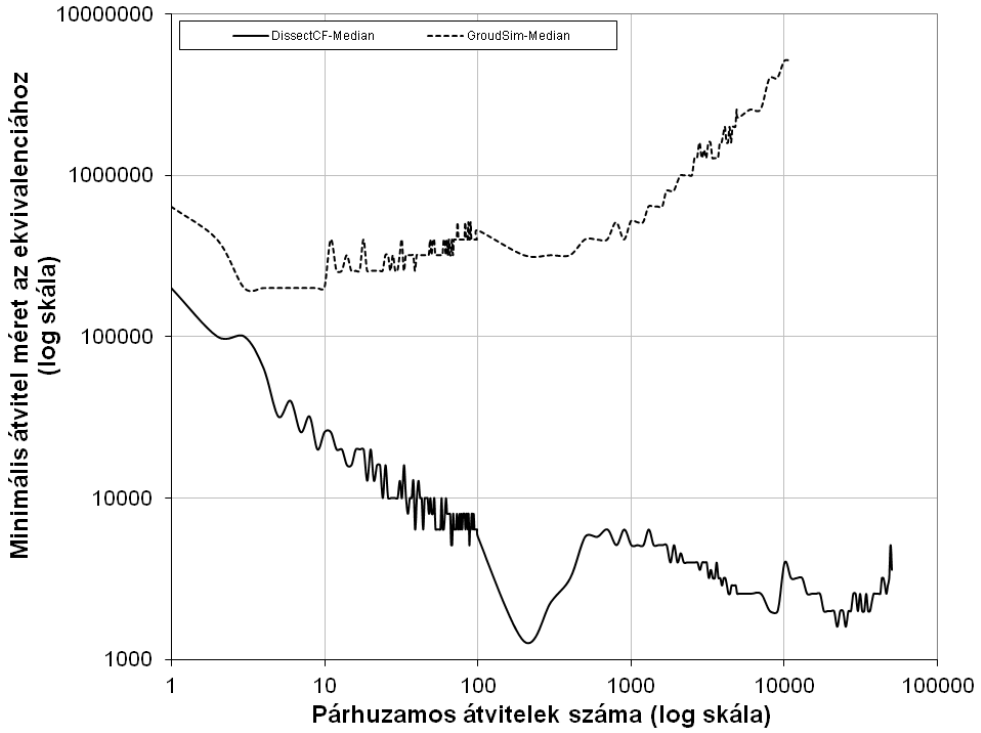
gezni (lehetősége van a hálózati átvitel sikertelensége esetén is külön aktivitásokat definiálni). Másodjára megkeresi melyik `Repository` tartalmazza a megfelelő fájlt amit abban a `Repository`-ban szeretne tudni amelyikben azt használni szeretné. Például, ha virtuális gép létrehozása során történik az átvitel akkor a forrás `Repository` a központi tár lesz ahol a másolandó VA elhelyezkedik, és a cél `Repository` pedig egy fizikai gép helyi lemeze. A kikeresett `Repository`-nak (az *1b.* lépésben) meghívja a `requestContentDelivery` függvényét amivel hivatalosan is elindítja az átviteli folyamatot. Ezzel a felhasználó szempontjából befejeződik az átvitellel kapcsolatos minden teendő, már csak várnia kell, hogy megérkezzen az *1a.* lépésben létrehozott eseménykezelőbe a teljesítést igazoló esemény.

A tényleges átvitel azonban még korántsem kezdődik el ilyenkor. Először is az átviteli kérést kapó `Repository` meggyőződik arról, hogy a fogadó `Repositry` megfelelő mennyiségű tárhellyel rendelkezik, ha igen lefoglalja azt. Ezután (az *1c*. lépésben) létrehozza a `NetworkEvent` továbbító objektumot ami majd az átvitel végén elvégzi a másolás befejezésével kapcsolatos teendőket, és értesíti a felhasználót. A továbbító objektumot megnevezve mint eseménykezelőt, a `Repository` felkéri a forrás és a cél `NetworkNode` objektumokat, hogy kezdjék el az átvitelt az `initTransfer` függvény segítségével (lásd *1d.* lépés).

A következő lépésben (*1e.*) a forrás `NetworkNode` objektum létrehozza a `SingleTransfer` objektumot (`X`) ami a majdani átvitelt fogja reprezentálni. Majd regisztrálja a létrejött objektumot mind a forrás mind pedig a cél `NetworkNode`-on (lásd *1f.* és *1h.* lépések). A regisztráció hatására a `NetworkNode`-ok megbecsülik a várható végét a hálózati átvitelnek és feliratkoznak (az *1i., 1g.* lépésekben) egy akkor érkező időzítő eseményre. Ezzel a lépéssel befejeződik az átvitelre való felkészülés és elindulhat az átvitel.

3.2.2. Átvitel

A DISSECT-CF szimulációk fő mozgatórugói az időzítő események. Ezek az események a szimulátor fő eseménykezelő ciklusából eredeztethetőek és ezredmásodperces idő felbontást tesznek lehetővé. Minden szimulációban kétféle eseményt különböztetünk meg: *(i)* idő függő és *(ii)* állapot függő. Az időfüggő eseményekre a `Timed` osztálynál kell feliratkozni, és az a megfelelő időpontban jelezni fogja, ha a kívánt esemény bekövetkezett. Ezekre az eseményekre iratkoztak fel a `NetworkNode`-ok is amikor átviteli kérés érkezett hozzájuk. A `Timed` osztály így aztán folyamatosan figyeli (lásd *2a.*és *2c.* lépések), hogy mikor kell az átvitel befejezéséről szóló időzitett eseményt kézbesíteni a csomópontoknak. A kézbesítés hatására a csomópontok a *2b.* és *2d.* lépéseknek megfelelően végrehajtják a `singleSpread` függvényeiket (először mindig az `OutputSpreader`, majd az `InputSpreader`). Ha más átvitel is folyik valamelyik csomóponton, akkor előfordulhat, hogy az hamarabb fejeződik be. Ekkor az időzített esemény kézbesítése kétszer történik, először a hamarabb befejeződő másik átvitelre, majd arra az átvitelre, amit az előző fázisbeli felhasználó kért. Amikor az átvitelt jelképező `SingleSpread` objektum már nem tartalmaz további átvivendő adatot, akkor az átvitel befejeződik és az időzített esemény átalakul állapot függő eseménnyé (amit a `NetworkEvent` továbbító fog megkapni).



Ábra 4 Valós mérés ekvivalencia

3.2.3. Átvitel befejezés

Amikor az *1c.* lépésben létrehozott eseménykezelőhöz megérkezik az átvitel befejeztét jelző állapotfüggő esemény, akkor az átvitel az utolsó fázisába érkezik. Ilyenkor kerül kiértesítésre minden érdekelt fél az átvitel végéről, és ilyenkor kerül lebontásra minden olyan objektum amelyet azért hozott a rendszer létre, hogy az aktuális átvitelt lemodellezhesse.

A `NetworkEvent` továbbító objektum első lépéseként (*3a.*) elvégzi a cél `Repository`-ban a másolt fájl regisztrálását. Majd a *3b.* lépésben értesíri a felhasználó által az átvitel megkezdése előtt létrehozott eseménykezelőt, hogy az átvitel sikeresen befejeződött. Ez az eseménykezelő megteheti a szükséges aktivitásokat, amik a sikeres fájlátvitel után teendőek, és persze elindíthat olyan nem hálózattal kapcsolatos aktivitásokat, amik ettől a fájlátviteltől függtek (lásd *3c.* lépés). Végül a hálózati szimulációért felelős komponensek elvégzik a saját karbantartásukat azzal, hogy a már elvégzett átvitelek regisztrációját megszüntetik (lásd *3d-e.* lépések) és ha szükséges a Timed osztálybeli időzitett esemény feliratkozásukat is törlik.

4. Hatékonyságvizsgálat

A következőkben egy egyszerű szcenárió segítségével összehasonlításra kerül egy DISSECT-CF hálózati szimuláció és a teljesítmény prioritással készült GroudSim [8]. A szcenárió a következő egyszerű lépésekből áll:

1. Készítünk 2 hálózati csomópontot

2. Létrehozunk N darab M byte hosszúságú átvitelt
3. Végrehajtjuk a szimulációját az összes átvitelnek
4. Lekérjük az szimulátortól eltelt szimulált időt T_{SZ}
5. Lemérjük a 2. és 3. pont között eltelt valós időt T_V

Minden egyes párhuzamossági szintre (N) meghatározzuk azt az átvitelméretet (M_M) amitől kezdve a szimulációban eltelt idő hosszabb mint a valós eltelt idő ($T_{SZ} > T_V | M = M_M$). Azaz amitől kezdve a szimuláció hamarabb eredményez értékelhető adatokat, mint egy fizikai tesztkörnyezeten felállított valós mérés.

Ezt a méréssorozatot mindkét szimulátorral ugyanazon az Intel(R) Xeon(R) X5570 @ 2.93GHz processzoron végeztem el, és az eredményeket az Ábra 5–en szemléltettem. Ahogy lehet látni, a DISSECT-CF nemcsak nagyobb teljesítménnyel képes a hálózati átviteleket szimulálni, hanem ugyanazzal a szimulációs környezettel (a 3.2 fejezet-nek megfelelően) képes a hálózati egymásrahatásokat is hatékonyan kezelni. A DISSECT-CF teljesítménybeli előnye ($M_{M\text{-}GroudSim}/M_{M\text{-}Dissect\text{-}CF}$) 3,6-as és 1600-as faktor között mozog, az előny mediánja a szimulált párhuzamossági esetekben 62,5-szörös.

5. Összefoglalás

Ebben a cikkben bemutatásra került a jelenlegi felhőszimulátorok egyik hálózatkezelési problémája. Erre a problémára megoldásként egy új szimulátor került kidolgozásra DISSECT-CF néven. A szimulátor hálózatkezelési eljárásait a cikk részletesen taglalta és bemutatta hogyan oldja fel a szimulátor a felvetett problémát is. A cikk azt is bemutatta, hogy a nagyobb pontosság mellett mégsincs szükség a komoly teljesítménygondokkal küzdő hálózati szimulátorokra jellemző csomagorientált szimulációk használatára. Végül a cikk bemutatta a DISSECT-CF hálózati szimulációs komponensének hatékonyságát egy könnyen hozzáférhető és sebességorientált szimulátorral (a GroudSimmel) összevetve.

Habár a szimulátor már jelenleg is használható, annak funkcionalitása még nem teljes (több adatközpontból épített felhőszolgáltatások, vagy felhő federációk támogatása még nem biztosított). Ezen felül a nagyméretű (több százezer processzort tartalmazó infrastruktúrákat is kezelni képes) szimulációk hatékonyabb támogatása érdekében többszálas végrehajtási modell kidolgozása vált szükségessé (jelneleg egy komplexebb szimuláció még pár ezer processzorral is több perces időintervallumot vesz igénybe, ami hátráltatja a DISSECT-CF-re alapuló nagyszámú szimulációs beállítást változtató szimulációk kipróbálási lehetőségét).

6. Köszönetnyilvánítás

A kutató munka az Ausztriai Kutatási Alap (Austrian Science Fund) TRP 237-N23 jelű projekt részeként és a Standortagentur Tirol RainCloud nevű projektjének támogatásával valósult meg.

7. Irodalom

[1] Varga, A. (2001). The OMNeT++ discrete event simulation system. *Proceedings of the European Simulation Multiconference* (old.: 1-7). ESM.

[2] Núñez, A., Vázquez-Poletti, J. L., Caminero, A. C., Castañé, G. G., Carretero, J. & Llorente, I. M. (2012). iCanCloud: A Flexible and Scalable Cloud Infrastructure Simulator. *Journal of Grid Computing* , 185-209.

[3] Garg, S.K. & Buyya, R. (2011). NetworkCloudSim: Modelling Parallel Applications in Cloud Simulations. *Proceedings of the 4th IEEE/ACM International Conference on Utility and Cloud Computing* (old.: 105-113). Melbourne: IEEE CS press.

[4] Casanova, H., Legrand, A. & Quinson, M. (2008). Simgrid: A generic framework for large-scale distributed experiments. *Tenth International Conference on Computer Modeling and Simulation* (old.: 126-131). Cambridge: IEEE.

[5] Fujiwara, K. & Casanova, H. (2007). Speed and accuracy of network simulation in the SimGrid framework. *Proceedings of the 2nd international conference on Performance evaluation methodologies and tools* (old.: 12:1-12:10). Brussels: ICST.

[6] Kliazovich, D., Bouvry, P. & Khan, S. U. (2012). GreenCloud: a packet-level simulator of energy-aware cloud computing data centers. *J Supercomput* , 1263-1283.

[7] Buyya, R. & Murshed, M. (2002). GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing. *Concurrency and Computation: Practice and Experience* , 1175-1220.

[8] Ostermann, S., Plankensteiner, K., Prodan, R. & Fahringer, T. (2011). GroudSim: An Event-Based Simulation Framework for Computational Grids and Clouds. In M. a. Guarracino, *Euro-Par 2010 Parallel Processing Workshops* (old.: 305-313). Berlin: Springer.

[9] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A. F. & Buyya, R. (2010). CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms. *Software: Practice and Experience* , 23-50.

[10] Henderson, T. R., Lacage, M. & Riley, G. F. (2008). Network Simulations with the ns-3 Simulator. *SIGCOMM'08* (old.: 527). Seattle: ACM.

[11] K Kurowski, A Oleksiak, W Pikatek, T Piontek, A Przybyszewski, and J Wkeglarz. DCworms–a tool for simulation of energy efficiency in distributed computing infrastructures. Simulation Modelling Practice and Theory, 39:135–151, December 2013.

[12] Michael Tighe, Gast´on Keller, Jamil Shamy, Michael Bauer, and Hanan Lutfiyya. Towards an improved data centre simulation with DCSim. In Proceedings of the 9th International Conference on Network and Service Management, CNSM 2013, pages 364–372, Zurich, Switzerland, 2013. IEEE.

[13] Ilango Sriram. SPECI, a simulation tool exploring cloud-scale data centres. In Cloud Computing, volume 5931 of Lecture Notes in Computer Science, pages 381–392. Springer, 2009.