Васил Георгиев Цунижев

МОДЕЛИ ЗА УПРАВЛЕНИЕ НА ОБСЛУЖВАНЕТО В КЛАУД

Хабилитационен труд за участие в конкурс за професор в Софийски университет „Св. Климент Охридски”

София
2013

Съдържание

Предговор 5
1. Архитектура и параметри на сервизната инфраструктура 9
2. Управление на ресурсите в клауд 22
2.1. Характеристика на ИТ-обслужването в клауд 24
2.2. Моделно пространство на ресурсното управление в клауд 29
3. Модели за управление на обслужването 47
3.1. Диаграми на състоянията и преходите на възлите 49
3.2. Случаи на балансиращите модели 52
4. Анализ на моделите за управление 55
4.1. Равновесни уравнения на моделите на възел 55
4.2. Анализ на параметрите на клъстера 57
5. Числов анализ на моделите 60
5.1. Числов метод за итеративно решаване на модела 61
5.2. Решаване на по-сложни модели с използване на агрегирани състояния 64
6. Зависимости между моделираните системни параметри 70
7. Обобщение 78
Терминологичен речник 81
Литература 85
Списък на цитирания 88

"I see all these services and more as we move headlong into the 21st century. Indeed, I foresee that the Internet will essentially be an invisible infrastructure serving as a global nervous system for the peoples and processes of this planet."

Leonard Kleinrock (2008, [13])

Основателно е да се доверим на легендарния създател на Интернет професор Клайнрок в неговата визия за развитието на глобалната информационна инфраструктура. Още през 1960-те, когато е проектирал IP-мрежите за американското военно ведомство, той е изказал пророческото предположение, че тази инфраструктура ще проникне до всеки потребител, като с това ще промени обществото и хората. Днес масовата персонализация, мултимедийното съдържание и всеобщата мобилност движат еволюцията на инфраструктурата и на изискванията към информационното обслужване. И това е еволюция, свидетелите на която считат по-скоро за революция. Съвременното развитие във всички информационно-технологични (ИТ) направления и най-вече в посочените три направления – персонализация, мултимедия и мобилност – се поддържа от платформи, структурирани във фóрмата на услуги (SOA – service oriented architecture) – фиг. 1.1.

Фиг. 1.1. Етапи на съвременната ИТ-еволюция.

Архитектурата на обслужване, която се базира на услуги, е развитие на софтуерните технологии (след обектното и компонентното програмиране) в посока към максимално едра грануларност. Едрата грануларност по принцип и особено при SOA означава технологичност, преносимост (поради високото ниво на софтуерната абстракция – напр. JVM), кратък проектен процес, взаимстване на код, бързо разгръщане на услугите и други предимства с технологическо и икономическо значение за пазара на ИТ-обслужването. Заедно с това обаче структурирането на архитектурата във фóрмата на услуги означава още висок системен свръхтовар и необходимост от сериозно развитие на инфраструктурата в три посоки – капацитет за локална обработка, капацитет на запаметяващите устройства и капацитет на свързаност. Тези изисквания допълнително се подсилват от масовизацията на достъпа до мултимедийно съдържание и други процеси с интерактивен (online) или реално-времеви режим.

С този преход – и като предпоставка, но и като следствие – е свързана еволюцията на инфраструктурата, която се обозначава с парадигмата клауд – т.е. разпределената сървърна инфраструктура. От системна гледна точка главните особености на клауд-инфраструктурата са следните:

- множество обслужващи сървери, които изпълняват съвместно набора от услуги на съответния доставчик (по-точно те изпълняват съвместно заявките за достъп до съответните услуги от страна на потребителите) в условията на частична или пълна преносимост на обслужването. Допълнителна особеност на съвременните сървери е, че по същество това са паралелни COMA-машини с многоядрени процесори; следователно комуникиращите услуги могат да извършват обмена или през локална обща памет или чрез обмен на съобщения – според моментното динамично алоциране на ресурси при заявка за достъп до съответните услуги; от това произтича и изискването за високо ниво на виртуализация, маскираща конкретния комуникационен модел[1];
- съществени за ефективността на инфраструктурата са системните услуги, които поддържат качеството на обслужване – например балансиране и разпределение на натоварването или на данните в клауда, а също и услуги на системно ниво, които разширяват приложната функционалност – например с:
    - поддръжка на директория за публикуване и достъп до услуги, до техния статус, до интерфейси към системи за търсене на услуги; достъпът може да е по модела Клиент-сървер, но също така все повече се налага и Абонаментният модел (publish/subscribe, pub/sub);
    - явно или неявно задаване на съставна суперуслуга чрез композиция (оркестриране/хореографиране) на няколко базови услуги с приложна и[ли] системна функционалност.

В този труд сме се постарали да обобщим група от модели, разработени с цел изследване и анализ на параметрите на съвременните начини за съхранение, пренос и интерпретиране на информацията. В крак с времето сме се фокусирали на клауд-

[1] при JVM комуникационният модел RMI се базира на обръщание към отдалечените обекти като към локални по модела обща памет чрез маскиране на поддържащия обмен на съобщения; при виртуалната машина на erlang точно обратно всеки обмен дори и между локални процеси с общо адресно пространство има синтаксиса на обмен на съобщения, с което се избягват страничните ефекти на общата памет.

инфраструктурата, на архитектурния подход, базиран на услуги, както и на достъпа до мултимедийни данни.

Изложението е развито в следната последователност. Първа глава представя кратък обзор на архитектурата и параметрите на сервизната инфраструктура. По-широко е развит преглед на подходите към моделирането и анализа на тази инфраструктура, без които е невъзможно тя да бъде оптимизирана спрямо конкретни функционални и нефункционални характеристики на обслужването. Втора глава обосновава системната архитектура и модела на обслужване, които ще изследваме. Трета глава представя група от стохастични модели, които описват процеса на обслужване от сервизната инфраструктура. Четвърта глава анализира основните зависимости между моделните параметри. В пета глава е предложен сходим числов метод, чрез който могат се анализират разработените модели. Шеста глава представя и обсъжда резултатите за характерни случаи на употреба на сервизната инфраструктура чрез анализ на моделните резултати за системните параметри. В заключение са направени изводи относно валидността и приложимостта на получените резултати, както и относно приложимостта и еволюцията на предложените модели и методите за тяхното анализиране. По същество този труд обобщава отделни публикувани резултати на автора, като ги свързва в единно изложение и ги разширява с общ обзорен контекст и приложна интерпретация.

1. Архитектура и параметри на сервизната инфраструктура

Съвременното информационно обслужване се развива в посока на универсализиран достъп и преход от документно към мултимедийно съдържание. Тази еволюция от една страна е следствие, а от друга е фактор за развитието на информационната сервизна инфраструктура. Тя изисква адекватно по темпове и качество адаптиране на поддържащите ресурси, за което възможните подходите са два – развитие на клиентското екипиране и на сърверното екипиране (включително и съответната свързаност чрез клиентските и опорните мрежи). Двата подхода се съчетават и допълват, но все пак принципите, на които се базират, се различават съществено. Нашето изследване е посветено именно на сърверните модели без да омаловажаваме ролята на развитието на клиентската инфраструктура: редица успешни бизнес-модели са базирани на равнопоставен обмен между клиентските процеси (peer-to-peer, p2p), макар че поради спецификата и проблемите на безсърверното обслужване все още p2p-обслужването не може да се разглежда като универсален модел, а и частично въвежда на йерархия между клиентите[1]. При съвременното развитие на информационното обслужване се оценява, че около 50% от обмена на данни се изпълнява вече извън опорните мрежи и свързаните в тях сървери [21].

Клауд-обработката е йерархичен модел на виртуализирано сърверно обслужване, при който информационно-технолигичните (ИТ) ресурси се групират прозрачно, като осигуряват по-ефективно изпълнение на заявките на инцидентните потребители или

[1] при протокола Skype например се различава функционалност на nodes и super-nodes [21].

въобще на клиентските процеси, поддържайки параметрите на обслужване в реално време или на интерактивно обслужване. Виртуализираните клауд-системи намират все по-голямо приложение, тъй като чрез тях се постига по-високо качество на информационнте услуги при обща тенденция на намаляване на стойността на проектиране, разгръщане и поддръжка на сервизната инфраструктура.

Клауд-системите се изграждат предимно с цел търговско предлагане на услуги (т.нар. продукционен клауд). В общия случай те съвместяват времево два основни типа обработка:

- обслужване на поток от асинхронни информационни заявки в изградените центрове за данни;
- изпълнение на потребителски програми с производителност и бързодействие, близки до тези на специализирана и резервирана сърверна инфраструктура.

От факта на съвместяване на тези две групи случайни заявки следва и решаващата зависимост на производителността и икономическата ефективност на клауд-обслужването от адекватен модел на ресурсното планиране, базиран разбира се на модел на натоварването – т.е. на стохастичния процес на постъпване на отделните заявки.

Освен икономическа ефективност, предимствата от клауд-обслужването произтичат и по отношение на предлаганата функционалност поради по-добрите възможности за групиране и композиране на услугите в суперуслуги посредством обръщение към системните услуги за наблюдение в клауда. Следователно клауд-виртуализацията позволява ефективно и прозрачно за потребителя – а и за доставчика на услуга – да се изпълняват традиционни сърверни приложения (cloud un-aware), а също и да се разгръщат специализирани клауд-приложения (cloud aware), които реализират част от функционалните си или нефункционални черти чрез поддържане на интерфейс към системните услуги на клауда.

Постигането на икономическа ефективност и функционална пълнота на сервизната инфраструктура чрез преход към виртуализирано клауд-обслужване изисква адаптиране и еволюция на традиционните методи за моделиране и оценка на производителността. Задачата на такова моделиране е да оцени в набор от подходящи параметри взаимовръзката между натоварването, инфраструктурните ресурси и параметрите на качеството на обслужване.

Така например моделирането на инфраструктурните ресурси трябва да отчете по-високата степен на споделяне на ресурсите между отделните услуги, която произтича от клауд-виртуализацията. При традиционното информационно обслужване, базирано на физически машини, основните ресурси, които се оценяват в интегриран вид или поотделно, са процесорна производителност, вътрешна и външна памет и мрежово свързване. При клауд-обслужването виртуалната сервизна среда предоставя принципно същите ресурси, но те са споделени между множество виртуални машини, които се намират в състояние на конкурентен достъп до физическите инфраструктурни компоненти.

В резултат на измененията при ресурсните модели се променя и моделът на натоварването. Принципно това се дължи на обстоятелството, че освен потребителите на услуги при клауд-обслужването натоварване се създава посредством системните заявки за споделяне на ресурси – т.е. от самия слой на виртуализация. Така при клауд потокът от заявки за обслужване е съставен от множество потоци от заявки, всеки от които може да има собствена динамика.

Възприети са различни подходи за моделиране на клауд-обслужването. Общото за тях е, че оценяват взаимовръзката между приложната производителност и разпределянето на ресурсите чрез представяне на системния вход и системния изход като подходящи набори от параметри, след което се обосновава и оценява функцията, която ги свързва. Тази свързваща функция най-често има вероятностен или статистически характер предвид природата на процесите на възникване на нови заявки за обслужване и на

напускане на обслужените заявки, както и на процесите на заемане, освобождаване, включване и изключване на ресурсите. Някои изследвания обосновават и подходи за анализ на управлението на клауд, различни от вероятностния и статистическия подход. Те използват например методи на изкуствения интелект и размитата логика. Не бихме се фокусирали тук върху изчерпателен обзор на методите за анализ на производителността на клауд, но ще посочим няколко принципни източника, всеки от които представя съответното методическо направление. По-надолу сме се спрели на техните общи характеристики и особености, а след това сме направили съпоставка на параметризацията на всеки от тях, както и на използваните съответни модели на натоварването.

В [20] Moschakis и др. представят ***стохастичен модел*** на обслужването в клауд в условията на групови заявки. Този модел е базиран на теорията на масовото обслужване. Като общи характеристики на моделираните системи, а и като изследователски метод той най-много се доближава до предложения от нас модел. Това впрочем е методът на анализ, развит и приложен от професор Клайнрок за разработването на протокола IP при зараждането на Интернет – съвременнта глобална информационна инфраструктура, а преди него – в нач. на XX век от датския математик Ерланг при проектирането на зараждащата се тогава глобална комуникационна (по-конкретно телефонна) инфраструктура.

Системата в модела на Moshakis и др. представлява хомогенен обслужващ клъстер от виртуални машини с централно планиране, базиран на платформата на Amazon Elastic Compute Cloud (EC2). Централното планиране (обща системна опашка) за хомогенен набор от обслужващи възли е модел, анализиран още в класическата монография на Клайнрок Queuing Systems I: Theory [14]. В модела на Moshakis обаче адаптирането към еластичния клауд на Amazon се изразява в две особености:

- възлите са виртуални машини с променлив брой, който зависи от алоцираните ресурси на доставчика на услугата и от интензивността на потребителските заявки;

- заявките може да са групови (т.е. множествени; тази характеристика на натоварването е изследвана също от Клайнрок, макар и в друг контекст). Да поясним, че под групова заявка в този модел се разбира заявка за обслужване, която позволява паралелно изпълнение на съответната потребителска функция (анализираният от авторите паралелизъм варира между 2 и 32).

При тези общи предпоставки авторите анализират със стимулационни методи своя модел в зависимост от различен модел на натоварването и в условията на действие на два централизирани планиращи алгоритъма: Adaptive first come first serve и Largest job first serve. Предимство на модела е и разширяването му с модел на ресурсите – т.е. анализ на включването и изключването на обслужващи виртуални машини – черта, която е характерна за обслужването в клауд. Да поясним, че на практика всеки модел на ресурсно управление включва явно или неявно и модел на натоварването, но примерно експерименталните модели могат да анализират реактивността на една реално функционираща система за обслужване и без да параметризират натоварването, тъй като разчитат на съществуващ поток от заявки за обслужване.

Моделът на Moshakis и др. е обоснован с обширен и добре фокусиран обзор от около 15 източника по специфичната тематика – т.е. други модели с подобни характеристики (без да броим останалите източници, които се отнасят към приложните технологии или към метода за анализ на модела). Поради изброените предимства и най-вече близост на изследователския подход, ние използваме именно този модел като база за съпоставка с нашите изследвания.

В [11] е представен ***статистически анализ*** на обслужването за продължителен период от време – една година – в продукционни клауд-системи съответно с платформите Amazon AWS и Google AppEngine. Статистическият анализ на експериментални данни дава особено ценни резултати преди всичко поради тяхната достоверност. Но основната слабост на този подход е, че може да се приложи едва след като изследваната сервизна инфраструктура е изградена в някаква значителна степен и е добила популярност сред

потенциалните потребители. Характерно за статистическия подход е, че макар по отношение на сервизната инфраструктура да се използва реално съществуваща и обикновено непрозрачна (когато е наета, а не собствена) инфраструктура, то по отношение на експерименталното натоварване на тази инфраструктура се създава модел на натоварването, който предписва с какви заявки и задания да бъде „атакувана" изследваната платформа. Самите експерименти при този метод се състоят в синтезиране (неправилно според нас наречено „симулиране" в [11]) на съответните заявки и измерване на времевите параметри на обслужването. В този смисъл тук входни параметри практически отсъстват – освен избор на платформа и продължителност на тестовете. Моделът на натоварването обаче позволява подробна параметризация.

Към конкретното изследване на колегите от групата на д-р Iosup може да се направи и забележката, че обработените данни са наблюдавани с времева стъпка от една седмица – в рамките на едногодишна или двегодишна продължителност. Така очевидно се губи представа за ежедневния и ежеседмичния цикъл на обслужване в продукционния клауд, а този тип кратки цикли също са важни именно поради глобалния характер на клауд-обслужването, който дава възможност за географско разпределение на заявките в рамките на едно денонощие или една седмица. Известно е например, че географското разпределение на трите центъра за данни на Amazon, е в три „диаметрално"-противоположни по времеви зони точки на Земята, което несъмнено има отношение към глобалния дневен цикъл на обслужваните заявки.

В [16] е описан анализът на модел на ресурсно управление, който се базира на итеративна обработка на информацията за наблюдаваната система с методите на ***самообучаващите се невронни мрежи***. В този модел се анализира хомогенна система от произволен брой изпълнителни сървери. Неявно се прави допускането, че тяхното централизирано управление може да обслужи произволно голяма система. Макар и твърде абстрактно, такова допускане всъщност е реалистично, тъй като то предполага само, че инфраструктурата трябва да се екипира с достатъчно производителен контролен сървер, съобразен с обработката на комбинирания свръхтовар от наблюдението и

управлението на изпълнителните сървърите в клауд-клъстера. Така анализът на произволна система се свежда до анализ на състоянието на един неин възел, закъсненията в който са представителни за цялата система. Този подход, който можем условно да наречем ***декомпозиционен подход***, е разпространен в редица анализи и обикновено е съпроводен с изискването за хомогенност на системата. Като правило клауд-клъстерите са именно хомогенни разпределени системи, следователно в общия случай декомпозиционният подход е приложим към тях без ограничения[1]. Друга особеност на модела в [16] е възприетият модел на натоварването. В този анализ системата на обслужване се оценява по времето за изпълнение на няколко еталонни алгоритъма (benchmarks), като обработката се изпълнява в монополен еднозадачен режим (т.е. без фоново натоварване). С цел да се изследва по-пълно системата за управление на обработката еталонните приложения са подбрани така, че да създават натоварване на различни подсистеми – интензивна процесорна обработка, алоцирана памет, интензивни входно-изходни операции и с комплексно натоварване.

След тази обща характеристика на всеки от така изредените модели на сервизната клауд-инфраструктура, ще се спрем на по-детайлна съпоставка на параметрите, които тези модели използват на входа и на изхода на моделиращия процес.

Параметризирането на даден феномен – в случая процеса на ресурсно управление – е основният и общ метод на всяко моделиране. Към така направения общ преглед на четирите различни моделни подхода би било полезно да се разгледа параматризиращата схема, с която се характеризира всеки подход. В крайна сметка именно набора от входни, междинни и изходни параметри на даден модел е неговата „азбука” или „език”. Съпоставката на параметризирането ни дава възможност да правим изводи по отношение степента на абстракция и адекватност на всеки моделен подход,

[1] Предварително ще отбележим, че нашите модели също анализират системните параметри именно съгласно този декомпозиционен подход. Все пак в предложените от нас модели се допуска декомопзиране на системата на повече от един клас виртуални машини, респективно всеки от тези (не много на брой) класове представлява група от виртуални машини с еднаква производителност. Класовете се анализират самостоятелно по принципа на декомпозицията, а на втори „глобален” етап систмата се анализира като съставена от краен брой класове.

като същевременно прозира принципното сходство между тях (нещо напълно логично при положение, че се изследва един и същ феномен макар и с принципно различен аналитичен апарат). Освен това чрез задаване на набора входни и изходни параметри по същество задаваме системата за наблюдение на инфраструктурата и съответно оптимизационните критерии за работата на контролните алгоритми. Междинните параметри по принцип са прозрачни извън модела, но могат да се разглеждат като потенциални входни или изходни параметри, поради което тяхното изброяване не е излишно (стига да са обявени). Ще разделим моделните параметри в две самостоятелни таблици – за модел на контролната система (Таблица. 1.1.) и за модел на натоварването (Таблица. 1.2.).

		A	B	C
	Модели / Параметри	Стохастичен модел ([20])	Статистически модел ([11])	Логически модел ([16])
1.	Тип на изследването	симулационно	експериментално	аналитично
2.	Входни параметри (приблизителен брой)	***6*** [вж. клетки A:2.1. ÷ A:2.7.]	***2***: периодичност на отчета и избор на наета услуга [вж. клетки B:2.1. ÷ B:2.7.]	***6*** [вж. клетки C:2.5. и C:2.7.]
2.1.	моделен период	общ брой задания N (в изследването N=64000)	1 година – седмичен или месечен отчет	произволен, кратен на броя обучаващи итерации на невронната мрежа
2.2.	размер на системата	$P_{\max}$ = 120 (брой виртуални машини)	Прозрачни за потребителя инфраструктурни и планиращи параметри съответно на Amazon Elastic Compute Cloud (EC2), Amazon Simple Storage Service (S3), Amazon Simple DB (SDB), Amazon Simple Queue Service (SQS), Google Run Service, Google Datastore Service, Google Memcache Service	произволен (принцип на декомпозицията)
2.3.	топология	звезда		всеки-към-всеки
2.4.	йерархичност	2 нива: планиращ сървер и $P_{\max}$ ВМ		2 нива: планиращ сървер и произволен брой ВМ
2.5.	планиращ алгоритъм	3 изследвани дисциплини: Shortest Queue First (SQF); Adaptive first come first serve (AFCFS); Largest job first serve (LJFS)		RPROP – итеративен тренираш алгоритъм за невронни мрежи (НМ тук са двуслойни с по 20 неврона във всеки слой) – 2 параметъра:

				активационна функция – с 3 възможни стойности (в това изследване) регресен модел – с 3 възможни стойности (в това изследване)
2.6.	наблюдавани ресурсни параметри	1 параметър: локалното натоварване на всяка ВМ		4 параметъра: CPU usage, memory usage, disk bandwith, network bandwith
2.7.	ниво на наблюдение на ресурсите	***3 нива*** = 2 прага на различими състояния на локалното натоварване – 1 и 2 задания в локално обслужване		***4 нива*** = 3 прага; напр. за алоцираната памет –100, 200 и 512 MB
3.	Междинни параметри	не са обявени		
4.	Изходни параметри (приблизителен брой)	7	9	1
4.1.	време на обслужване	5 стойности: ART (Average Response Time), AWRT (Average Weighted Response Time), AWT (Average Waiting Time), AWWT (Average Weighted Waiting Time), ABSLD (Average Bounded Slowdown), APPSLD (Average Per-Processor Slowdown)	3 стойности: Average Response Time (ART); Query Response Time; Update Latency; Average Lag Time;	
4.2.	скорост но обмена		2 стойности: Get Throughput; Put Throughput	
4.3.	системен свръхтовар	измерва се с цената на обслужване – тук Total Lease Time (LT)	3 стойности: Deployment Latency; Average Bounded Slowdown (ABSD)	Training and Deployment Overhead (измерва се в секунди закъснение за обучение на невронната мрежа)
4.4.	ефективност	Cost-Performance Efficiency (CPE) – комбинира ART и LT	Aggregate Performance Penalty	

Таблица 1.1. Съпоставка на моделите на контролната система

		D	E	F
	Модели / Параметри	Стохастичен модел ([20])	Статистически модел ([11])	Логически модел ([16])
1.	Параметри на модела на натоварването (приблизителен брой)	4	5	1
2.	процес на постъпване на заданията – интензивност	експоненциално разпределение на единен входящ поток със средна стойност $1/\lambda$	среден брой потребители (relative number of users) ~10^2 и среден брой задания за всеки ~10^3/годишно	
3.	сложност	измерва се с паралелизъм на всяко задание: $[1..16]$ с вероятност q и [17..32] с вероятност 1-q	Average Response Time (ART) – разпределено по два сценария	измерва се с време за изпълнение на еталонен алгоритъм: Sysbench-CPU, Memorymicro-benchmark, PostMark (интензивен В/И), OLTP (комплексен)
4.	грануларност	3 нива по вероятността q: 0.25 (едра), 0.50 и 0.75 (фина)	по-едра от 10 секунди: average bounded slowdown (ABSD) > 10 S	
5.	процес на напускане на заданията	експоненциално разпределение на изходящ поток със средна стойност $1/\mu$ за всяко подзадание на паралелните задания	като Е.3. – по 2 сценария	

Таблица 1.2. Съпоставка на моделите на натоварването

В резултат от направената съпоставка можем да направим следните изводи.

Най-развита е параметричната система на симулационния стохастичен модел A./E. ([20]). Това се дължи преди всичко на неговата формалност. При аналитичните и симулационните модели изследователят е свободен да дефинира параметри за всеки интересуващ го феномен – включително и междинни параметри, позволяващи да се коригира и наблюдава самият процес на оценка на модела. При решаване на модела именно стойностите на междинните параметри са индикатор за достоверност на резултатите, които ще се получат на изхода на моделирането. Трябва да се има предвид,

че тези междинни параметри имат обикновено съвсем конкретен смисъл и именно поради това могат да служат като коректен индикатор за достоверност на моделните резултати или за отсъствие на грешки в анализа. За пояснение ще дадем следния пример. Междинен параметър в [20] би могла да бъде вероятността за нулева дължина на локалната системна опашка за активни процеси в даден сървер от обслужващия клъстер. Първо като вероятност този параметър може да получава съвсем ограничени стойности – в интервала [0, 1]. И второ: съществуват съвсем практични съображения от типа на следното: при високо натоварване на системата не е логично да получим високи стойности на този междинен параметър – близки до 1 (по-вероятно е обратното – твърде рядко сървърът ще остане в състояние празен, съответно вероятността за нулев брой процеси в обслужване ще клони към 0). Трябва да отбележим, че при всички цитирани тук изследвания – включително и [20] – отсъстват данни за междинни параметри. Очевидно причина за това е преследването на пестеливост на изложението. Но по принцип именно аналитичните и симулационните модели позволяват най-развита параметризираща схема. Това наблюдение е валидно и по отношение на параметризиращия модел на натоварването.

Съпоставима по пълнота е и параметризиращата схема на експерименталния статистически модел. Все пак най-същественото предимство на експерименталния подход е неговата достоверност, постижима обаче само a’posteriori – т.е. когато изследваната система не само съществува, но и се използва с определена интензивност. Друга особеност в този случай е, че моделът на натоварването може да се настройва само в определени граници – независимо какви тестове и праметризация прилагаме – тъй като обикновено се изследва инфраструктура, която е в процес на обслужване на потребители. В повечето случаи изследователят не разполага с данни за това фоново обслужване на потребители (така е и в [11]): не може да се регистрират моментите на постъпване на заявките, тяхната сложност, кратност (т.е. паралелизъм). Тези важни за общия модел параметри на обслужването могат да бъдат оценени само косвено чрез регистриране на параметрите на обслувжане на специално генерираното натоварване – т.е. поток от

заявки – при експериментите. Друг експериментален подход – приложим обаче само в случай, че изследването на производителността се прави от администратора/собственика на инфраструктурата или услугите – е да се регистрират именно моментите на постъпване и напускане на потребителските заявки в клауд-инфраструктурата.

Лансираните в нашето изследване модели се отнасят към групата на стохастичните модели с аналитично решение. Избраният моделен апарат, както и разработеният метод за решаване на моделите, позволяват най-разгърната параметрична система, включително междинни параметри, позволяващи контролиране на достоверността на цялостния моделен процес.

Важна характеристика на моделирането на СВ е ***скаларността***. При скаларните модели на ресурсите и на натоварването всяко задание представлява статистически идентично компютърно натоварване, описано със скаларна стойност – като негова абстрактна времева сложност – например брой елементарни операции. Освен това тези операции биха се изпълнили за едно също време от всеки сървер на клъстера при условие, че няма фоново обслужване. Мнозинството модели на СВ са скаларни – [19, 31, 32, 3, 1, 26 и 27].

За сравнение ще посочим и ***векторни модели*** на натоварването и ресурсите. При тях заданията, както и параметрите на изпълняващите ги възли, се описват от [малко] множество атрибути на обработката. Най-често по отношение на модела на натоварване това са броят операции, размерът на алоцираната памет за изпълнение на заданието, броят операции към външна памет или входно-изходно устройство [30, 16]. Клетка 2.6.C. на табл. 1.1. илюстрира точно такъв ресурсен вектор по отношение на модела [16]. [20] пък има векторно представяне на своя модел на натоварване – вж. клетка 3.D. на табл. 1.2. В този модел скаларната времева сложност на отделните задания е разширена с параметъра паралелизъм и допълнително с вероятност за възникване на заявки за паралелно обслужване от група услуги, или (ако се съди по потребителския сценарий на [20]) по-скоро от група реплики на една услуга.

Нашият коментар по отношение на саларност/векторност на моделите за СВ трябва да изхожда от позицията на практичност, приложимост и адекватност на тази черта – от една страна – но също и от позицията на аналитичност на съответния модел. По същество това е проявление на баланса между детайлност и абстрактност, който е характерен за всеки моделен процес.

Както ще видим в секция 2.2. лансираният от нас модел е скаларен, тъй като преценяваме, че множеството ресурсни и товарни параметри могат да бъдат преизчислени към един параметър – времето за обслужване (или броя абстрактни операции). Единственият параметър, който не може да се оцени пряко във времеви мерни единици поради неговата ортогоналност, е заетата памет. Това означава, че се абстрахираме от въпроса за балансиране на натоварването освен по времето за обслужване и според необходимата памет.

В допълнение държа да отбележа, че някои модели не могат да бъдат скаларни по начало и при тях трябва да се разработи по-сложният за анализ векторен вариант. Пример за такъв модел е т.нар. зелен клауд [2], където оптимизационен критерий освен производителността е и екологичната цена на обслужване, измерена например чрез разхода на електрическа енергия. Всъщност векторен би бил всеки модел, който поставя въпроса за цената на обслужването – и в класическия паричен смисъл също.

2. Управление на ресурсите в клауд

Виртуализацията и управлението на ресурсите в клауд са главните системни функции на това ниво на платформена абстракция. Виртуализацията традиционно се изразява в имплементиране на единен интерфейсен език към различни системни услуги – в т.ч. и системни услуги от различни слоеве на изпълнителната платформа. Трябва да отбележим, че виртуализацията е значително по-сложен проблем за решаване при грид-обслужването, отколкото при клауд. Главната причина за това е, че по принцип грид-инфраструктурата обикновено представлява федериране на ресурси на няколко административно разделени организации в една виртуална организация. В резултат гридовете обединяват хетерогенни ресурси и съответно виртуализацията им до единен интерфейсен език за достъп е една наистина трудно поддаваща се на стандартизиране задача. В известен смисъл именно осъзнаването на виртуализационните проблеми при гридовете породи като реакция бизнес модела на масово обслужване в клауд, който в този контекст може да се опише и като хомогенен грид с централизирана администрация.

При клауд-обслужването поради единното администриране на ресурсите – включително и на етап разгръщане и надстройка на инфраструктурата – виртуализацията е значително по-лек проблем и може да се разглежда по-скоро като инженерно проектиране, отколкото като изследователска задача. Все пак обаче виртуализационните протоколи представляват слой от протоколния стек на разпределените клауд-системи и

следователно техните нефункционални характеристики участват като параметър в общия модел на обслужване.

Централната изследователска тема при клауд това е модела на управление на ресурсите. Понеже – пак за разлика от обичайния потребителски сценарий на грид – клауд-обслужването се извършва в условията конкурентен пазар на услугите, ефективната организация на ресурсите става решаващ компонент в цената на услугата. Задачата на управлението на ресурсите в клауд от една страна е да се повиши ефективността на инсталираната инфраструктура и на качеството на обслужване, но също така да се идентифицират тесните места и съответните мерки по еволюция на апаратния и системния ресурс. Системите, осигуряващи качеството на обслужване в клауд, са от изключителна важност за приложимостта на този модел на обслужване. Освен бързодействие, тяхната задача е да се гарантира самият достъп до услугите

Основен и традиционен подход при управлението на ресурсите е балансирането на изчислителния товар – проблем, който в контекста на клауд обслужването често се означава като cloud balancing (CB) [19]. Понятието „балансиране" в случая е леко подвеждащо, тъй като задачата на CB всъщност не е всеки от сървърите в един клауд-клъстер да изпълни еднакъв брой заявки за обслужване или пък примерно еднакъв общ брой операции, а по-скоро във всеки момент на обслужването да няма сървър в състояние на изчакване на заявки, докато в същото време друг сървър отлага изпълнението на постъпили заявки в съответствие с локалния си планиращ процес и по този начин се влошава качеството на обслужване и ефективността на инфраструктурата като цяло.

Едновременно с проблема за ефективно и качествено обслужване, CB обикновено има за задача да постигне и отказоустойчивост чрез прехвърляне изпълнението на сервизни процеси от отказал към работещ сървър [28]. По принцип отказоустойчивостта е качество на обслужването което се постига чрез репликиране на услугите и следователно имплементирането на тази черта се осъществява изцяло в рамките на балансиращите

натоварването системни процеси като използва тяхното планиране и наблюдение. Инженерно погледнато чрез репликиране на услугите се имплементира и схема за възстановяване след грешка [32].

2.1. Характеристика на ИТ-обслужването в клауд

Нека отбележим кои са съществените особености на обслужването в клауд [7, 27].

Режимът на обслужване в клауд обикновено се описва като ***асинхронна обработка*** на отделните задания с паралелизъм по данни. Това е най-благоприятният режим на работа за осъществяването на ефективен СВ, тъй като дава най-голяма свобода на съответния планиращ процес. Търговски-ориентираната клауд-инфраструктура чрез ефективния си СВ представлява икономична алтернатива на разширяването на информационното обслужване като може да замести необходимостта от добавяне на нови центрове за данни. Центровете за данни изпълняват асинхронни заявки за обслужване на потребителите, като изпълнението на дадена заявка е независимо по време и възел от това на останалите заявки – модел, който очевидно дава много по-голяма свобода на планиращия процес в сравнение с модел, при който част от заявките са обвързани по време и/или място на изпълнение.

Изключение от принципа на асинхронно обслужване в клауд е сценарий, при който потребителят използва самата инфраструктура като услуга и то за изпълнение на ***паралелна обработка***, съставена от множество синхронни процеси. Показателно е, че точно този модел на обслужване или не се предлага от доставчиците на клауд-услуги или струва по-скъпо от асинхронното обслужване именно поради значителното намаляване на ефективността на СВ, който е ориентиран към планиране на асинхронни заявки. Например свободните (неплащащи) потребители на Google Engine могат да зададат изпълнението на група от задания в инфраструктурата на Google, но при това не могат да укажат тяхното съвместно планиране (co-sheduling, т.е. едновременното им изпълнение) и следователно не могат да изпълняват безплатно синхронни паралелни задачи.

По отношение на асинхронното обслужване в клауд трябва да посочим обаче едно много важно изключение – ***мултимедийните данни***. Както посочехме в гл. 1. – мултимедийните данни все повече се превръщат в основна форма на съхранение и представяне на информацията. А интерактивният достъп до мултимедийни данни (с непосредствено възпроизвеждане при клиента и евентуален контрол на сървера от типа на старт-стоп-напред-назад[1]) както е известно изисква изосинхронен режим на предаване на последователните пакети. От гледна точка на сервизната инфраструктура – т.е. като се абстрахираме от клиентската интерпретация – мултимедията представлява поточни данни (streaming). Това означава, че вместо типичния за клауд модел на обслужване на поток от асинхронни заявки, в случаите на интерактивен достъп до мултимедия инфраструктурата трябва да осигури последователно предаване на множество пакети през равни времеинтервали със сравнително малък допуск за отклонение от графика. С други думи макар единичната заявка за достъп до мултимедия да се вписва в общия модел на асинхронно обслужване на множество потребители, то нейното изпълнение ангажира задълго ресурс по определен график, зависещ от характеристиките на съобщителния канал – които впрочем са динамични и могат да варират по време на изпълнение на такава заявка. Отсега ще отбележим, че предложените от нас модели могат да бъдат анализирани и за такъв тип натоварване, но тук не представяме такъв анализ.

Клауд-системите обикновено се състоят от един или повече специализирани центрове за данни, разделени географски в съответни стратегически точки. Когато обслужването е глобално по потребителски обхват, географското разпределяне позволява планиране на заявките според часови зони, дни от седмицата и даже климатични сезони, тъй като това са фактори, влияещи на ритъма на потребителските заявки и следователно на модела на натоварването. От друга страна факторът мрежова свързаност губи своето значение поради все по-нарастващата глобална свързаност поне що се отнася до местата, където се разполагат центровете за данни. Така например според цитираният източник [28] глобалното обслужване на портала Amazon се състои от три центъра за данни,

[1] най-масовия пример за такъв е моделът на You Tube.

разположени съответно в Япония, Северна Ирландия и на източното крайбрежие на САЩ, към които през 2011 – 2012 г. са добавени четири нови, следващи принципа на географско „балансиране” – фиг. 2.1. [29].

Фигура 2.1. Разположение на центровете за данни на Amazon.

Центровете за данни се изграждат като сървърни ферми – т.е. клъстери от сървери с оптимизирана системна специализация (за данни, за обмен, контролни и т.н.). Следователно ***клъстеризацията*** е друга особеност на обслужването в клауд. Тя позволява практически неограничено ескалиране на обслужващия процес както по отношение на надварването, така и по отношение на информационния ресурс. Принципът на клъстеризация е приложим не само към инфраструктурата като цяло, но и към отделните й съставни компоненти – например контролните процеси във фермата на даден център за данни могат да се изпълняват не от един, а от няколко контролни сървера, формиращи нейният контролен клъстер. Следователно по отношение на модела

на СВ изводите от тази черта на клауд-обслужването е, че този модел трябва да е съобразен със следните изисквания:

- да прилага разпределена схема на контролните процеси;
- разпределението на контролните процеси да дава възможност за свързване и управление на заявките за обслужване между потенциално повече от един центрове за данни (т.е. клауд-клъстери); дали тази възможност се реализира от равнопоставени комуникиращи контролни процеси (peer-to-peer, p2p) или чрез допълнителен йерархичен междуклъстерен слой е проблем, който може да бъде анализиран по отношение на ефективна приложимост чрез резултатите от моделирането;
- логично е концепцията за обединяване на клауд-клъстерите в единна сервизна инфраструктура да са развие в концепция за интер-клауд обединение. Но предвид административната фрагментарност на такова обединение възниква въпросът за стандартизирането му. Но на този етап не съществува технологична зрялост за въвеждане и прилагане на единен стандарт за интер-клауд интерфейс [19][1].

Клауд-системите са ***хомогенна платформа***. Хомогенността се изразява по отношение на различените протоколни слоеве или услуги – IAAS, PAAS и SAAS. Тя е и една от основните отличителни черти на клауд спрямо модела на обслужване в грид (където се налага виртуализация между няколко различни протоколни стекове поради различната административна принадлежност на ресурсите). Хомогенността в клауд произтича от единната административна принадлежност на цялата сервизна инфраструктура, а също и от това, че като правило инфраструктурата е изградена на един сравнително зрял етап на технологичното развитие и е специализирана именно за клауд-обслужване (т.е. не се

[1] "The intercloud is a lot like the set of all clouds connected via standards-based mechanisms. What those mechanisms are may be up for discussion and there are certainly groups devoted to defining those mechanisms but suffice to say that right now the "intercloud" does not exist. It (probably) will but we're a ways off from that"

налага обединяване на ресурси с хетерогенни контролни слоеве както при грид). Хомогенността рефлектира върху контролния модел в следното:

- еднакви схеми на управление във всички сървърни възли и евентуално във всички клъстери – това се отнася най вече до схемите на системния процес на наблюдение на ресурсите и заявките за обслужване;
- опростен процес на преместване на обслужването между сървери и евентуално клъстери – с цел балансиране/ефективност (или с цел отказоустойчивост) – а това има съществено значение при анализа и оценката на системния свръхтовар на дадена СВ-схема.

Съществена характеристика на параметрите на клауд-обслужването е тяхната ***динамичност***. Пазарната конкурентост на клауд обслужването изисква динамично ресурсно скалиране [1]. Впрочем тук понятието "цена" може да е съставено от множество компоненти освен паричната стойност – най-очевиден допълнителен компонент е екологичната цена на обслужването, с която се занимава изследователският отрасъл на „зеления" клауд [2]. Добър пример за динамичност на ресурсите и на параметрите на обслужване е „еластичността" на клауд-обслужването на Amazon, която отразява именно динамичността на обслужващия капацитет на техния клауд и се променя в рамките на минути, а не примерно на секунди или пък на часове и дни [28, 29].

Клауд-системите се състоят от няколко слоеве – обикновено три. Такова разслояване съответства на управлението на различните модели на обслужване:

- комуникации и сървери – за IAAS;
- приложения и информационни масиви – за PAAS;
- услуги и суперуслуги – за SAAS.

Следователно пълният модел на управление на ресурсите би трябвало да е също съставен на принципа на ***разслояването***: да се базира на наблюдение на ресурсите от различните системни слоеве, а също така и на слоест модел на натоварване (т.е. на различените категории заявки).

Обобщавайки тези черти на клауд-архитектурата, виждаме, че за да се постигне пазарна конкурентост на качеството на обслужване в такава отворена публична система управлението на ресурсите трябва да прилага разнообразни оптимизационни критерии – т.е. разнообразни балансиращи натоварването стратегии. В резултат от това всеки адекватен модел на СВ има моделно пространство, което се характеризира с многомерност и с широк обхват на отделните параметри. Следователно едно изчерпателно параметрично изследване на такъв модел предполага и допълнително изискване по отношение на изчислимост – за да могат да се съпоставят стойностите на наблюдаваните изходни параметри за голям брой комбинации от стойности на входящите параметри.

Нека сега изложим основните характеристики на лансираните от нас модели на обслужване в клауд.

2.2. Моделно пространство на ресурсното управление в клауд

Комплексният модел на обслужване в клауд-инфраструктура се състои от модел на ресурсите, модел на натоварването и модел на управлението.

2.2.1. Модел на ресурсите и на натоварването.

Съгласно описаното в предходната секция, приемаме, че СВ се изпълнява в е д и н хомогенен сървърен клъстер. Засега оставяме настрана въпроса за междуклъстерното управление. Стартирането на услуги се предизвиква от процес на асинхронно постъпващи независими единични заявки. Следователно друго ограничение на представяния модел е по отношение възможността за изпълнение на паралелни потребителски приложения например от типа на суперуслуги, композирани от потребителя на базата на елементарни услуги. Това ограничение обаче не касае възможността дадена услуга да бъде сама по себе си паралелно или разпределено приложение. Определянето на изпълнителен възел или друг ресурс на клъстера, който да стартира така заявената услуга, става динамично и

конкурентно на останалите услуги, изпълнявани в дадения момент от клауд-клъстера. Понятието „конкурентно” в случая има тесния специализиран смисъл „паралелно (т.е. в един мултипроцесорен сървърен възел) и л и разпределено (т.е. в даден отдалечен възел, определен според клъстерния контролен алгоритъм за СВ).

Моделът на ресурсите и на натоварването е ***скаларен***. Всяко задание представлява статистически идентично компютърно натоварване, описано със скаларна стойност – като негова абстрактна времева сложност – например брой елементарни операции. Освен това тези операции биха се изпълнили за едно също време от всеки сървър на клъстера при условие, че няма фоново обслужване. По време на обслужването постъпващите заявки за достъп до услуги съставляват стохастичнен процес на постъпване, докато изпълнените услуги формират статистически процес на напускане. Както беше отбелязано в края на секция 1., скаларният модел в този случай означава абстрактност по отношение на:

- ресурса памет – т.е. допускане, че сървърите имат достатъчно памет;
- цена на обслужването във всякакъв друг смисъл освен времеви – електроенергия, парична стойност и т.н.

Все пак нека отбележим отсега, че преход от скаларен към векторен модел не е невъзможен, а освен това скаларните модели може да предоставят междинни резултати, които позволяват оценка на множество параметри, т.е. решават задачи, съпоставими по комплексност с тези на векторните модели. Например в нашия скаларен модел основният оценяван изходен резултат и средното време за изпълнение на услуга, но ползвайки междинния резултат за среден брой заявки в процес на обслужване във възел и в клауд-клъстера очевидно може да се оцени необходимият ресурс памет – по различните йерархични нива от слоевете на кеша към външната памет – за възел и за клъстера. От вероятностното разпределение на броя обслужвани заявки може да се оцени също така и вероятността за препълване съответно на процесорния кеш, на междупроцесорния кеш, на външния кеш и на основната памет. За тази цел е необходимо, разбира се, да се познават реалните параметри на заявките за обслужване.

Фигура 2.2. Системни услуги на баланисращия натварването процес.

За да разработим относително сложния модел на СВ, е необходимо да декомпозираме този контролен процес на три съставни системни функции, които по традиция се наричат ***стратегии*** (strategies [4], в някои източници се срещат и като policies[15, 18]). В SOA всяка функция – включително и системните функции – се структурира като услуга. Следователно за конкретност тук ще използваме именно понятието „услуга" за платформените имплементации на тези три стратегии. А те са услугата за наблюдение, услугата за алоциране и услугата за трансфер на заявките [4] – (фиг. 2.2.). Това са три съвместно допълващи се процеса със различна праметризация и сложност. По принцип услугата за наблюдение е определяща по отношение на другите два процеса, тъй като резултатът от нейното изпълнение се явява входен аргумент на услугата за алоциране на ресурси за постъпилите заявки, а съответно решението на задачата за алоциране се изпълнява от услугата по трансфер. Често от изпълнителните процеси за алоциране и трансфер има обратна връзка към услугата за наблюдение, т.е. зависимостта между тях може да е циклична. Но трябва да се има предвид че въпреки тази взаимна обвързаност между трите процеса, те все пак могат да бъдат базирани на

различни принципи [30]. Досататъчно е да отбележим, че всяка от тях независимо от останалите две може да бъде структурирана като

- един централизиран процес по модела клиент-сървер (CS);
- няколко йерархични процеса по различни йерархични архитектурни шаблони;
- самостоятелен процес във всеки възел на клауд-клъстера по модела всеки-към-всеки (p2p).

Така например централизирано наблюдение може да бъде имплементирано на един сървер в клъстера, който наблюдава натоварването и състоянието на ресурсите във всички възли, а събраната от него информация може да се ползва от разпределен локационен процес, който се изпълнява от всеки възел.

Както обичайно при дилемата между централизирани и разпределени архитектури така и при стратегиите за СВ компромисът е между простота, технологичност и модулност на централизирания модел с по една услуга за трите балансиращи функции и напълно разпределения p2p модел, характеризиращ се със скалируемост и отказоустойчивост, но с по-голяма сложност на имплементацията и поддръжката. Така е първата характеристика на трите балансиращи услуги е тяхната ***контролна схема*** т.е. модела на разпределяне, който – както виждаме – може да е различен за всяка от тях.

При ***централизирана контролна схема*** един клъстерен контролер изпълнява някои или всички от функциите на СВ – наблюдение, алокация и трансфер. Това е разпределена архитекура от типа клиент-сървер ***(CS)***, при която сървърът наблюдава ресурсите и заявките за обслужване, като с това обикновено поддържа някалака системни функции освен СВ – например отказоустойчевост и самовъзстановяване след грешка, кампютърен одит и защита на информацията и др. т. Други системни (а всъщност и оторизирани потребителски) процеси изпълняват своите функции на базата на данни от сървера за наблюдение. Алоцирането и трансферът също могат да бъдат централизирани в съответни сървърни процеси. CS-схемите се асоциират с директен обмен на задания:

поради централизираната процедура вероятността за пре-предаване на едно задание от донор към приемник, който междувременно да стане донор за друг приемник, е много малка, а може и да бъде изключена по принцип от самия балансиращ протокол.

Централизираният балансиращ процес обикновено се базира на доста прибилизителна (като точност на стойностите и като честота на наблюдение) информация поради загриженост за скалируемастта и системния свръхтовар [31]. Освен неточност на наблюдението, ако се разчита само на централизирано балансиране друг важен фактор за товарен дисбаланс между обслужващите възли е непредсказуемастта на времето за обслужване на отделните задания [12]. Поради това в различни публикации [7, 10, 8, 9, 27] сме предлагали децентрализация на контролните схеми в различни степени и форми, които ще систематизираме по-долу тук и в гл. 3.

Децентрализацията на контролните схеми може да бъде пълна или частична. ***Глобално разпределената контролна схема*** може да се разглежда като приложение на модела всеки-към-всеки ***(p2p)***. Всеки възел изпълнява тройката контролни процеси локално и преговаря с в с и ч к и останалите контролни процеси в клъстера балансиращия обмен на задания. За разлика от CS-схемите, при p2p вероятността от итеративни балансиращи трансфери на задания не е пренебрежима и не би трявало да се изключва по принцип, тъй като балнсиращите решения се вземат локално. В идеалния вариант при глобален p2p всяка локална услуга за наблюдение, на базата на която се вземат решенията за балансиращ трансфер, би трябвало да разполага с актуална информация за натоварването на всеки от останалите възли. На практика, разбира се, са възможни отклонения и противоречие между данните в различните възли. Друга особеност на разпределените контролни схеми е, че поради по-сложния контролен механизъм, съставен от съвместното действие на множество контролни процеси, някои от трите системни процедури на балансирането могат да бъдат опростени.

Един пределно-опростен вариант на глобално p2p балансиране е изобщо да се изключи процедурата по наблюдение, която е и най-„тежката”, свързана със множество

операции по обмен на съобщения – и то от тип общодостъпно предаване. В такъв случай при настъпване на събитието претоварване – т.е. когато възелът мине в състояние донор – новопостъпили заявки за обслужване се препращат към произволен друг възел от клъстера. Такава локационна процедура може да изглежда твърде примитивна, за да бъде ефективна, но има смисъл да се анализира най-малкото като контролен граничен случай. Освен това такъв тип контролни алгоритми, известни с названието „горещ картоф”, намират практично приложение при някои отделни случаи на маршрутизиране в телекомуникациите – следователно тяхната ефективност не бива да се изключва априори. В нашето изследване разглеждаме този граничен случай на p2p балансиране и го означаваме като ***Браунов*** (поради прилика с хаотичното молекулно движение при високотемпературните флуиди).

При ***локално-разпределените контролни схеми*** (***Lp2p***) за разлика от глобалните се дефинира релация на съседство между възлите – по-точно между техните балансиращи процедури – при което мониторинговият и трансферният обмен всеки-към-всеки е само със ограничен брой „съседи”, вместо с всички останали възли в клъстера; съответно комуникациите не са общодостъпни, а групови. В резултат на тази релация на съседство балансиращите процеси на възлите формират свързан граф и неговите параметри – топология, валентност на възлите (бр. съседи), диаметър и т.н. – се явяват параметри на балансиращите контролни процеси. В предложените тук модели означаваме Lp2p-схемите като ***дифузионни*** – отново по аналогия с термина от молекулярната динамика – тъй като при тези схеми се допуска пре-предаване на задания и мониторингови съобщения.

Понятието за дифузия е въведено в контекста на управление на натоварването в разпределените системи в моделите на George Cybenko [3]. Той изследва характеристиките на разпределена система, която не поддържа общодостъпно и групово предаване, а вместо това възлите са свързани в мрежов граф с определена топология – по-конкретно в това изследване мрежата е хиперкуб. В [12] същата концепция е обобщена за произволна топология (зададена с матрица на съседство) на

комуникационната мрежа – но също без поддържане на общодостъпно предаване. В контекста на клауд-клъстерите – които, разбира се, поддържат общодостъпно предаване[1] – тези модели представляват интерес именно с оглед на въвеждане на ограничения върху общодостъпното предаване чрез дефиниране на релация на съседство с определена топология между възлите. По този начин се цели редуциране на системния свръхтовар, тъй като обхватът на действие на разпределените процедури за наблюдение и алокация се стеснява само до съседите на конкретния възел според така формираната мрежова топология. Например такива са топологиите линия – с един съсед, пръстен – с две съседа, d-дименсионна решетка – с $2d$ съседа, d-дименсионен хиперкуб – с d съседа и др. т.

Друга важна характеристика, която се отнася основно към услугата за наблюдение, е ***прагът на точност*** на събираната системна информация, въз основа на която се прави балансиращото планиране. При СВ можем да характеризираме локалното натоварване на всеки възел в даден момент с броя заявки в процес на обслужване – и това е параметър на лакалното планиране на ресурсите. На клъстерно (т.е. на клауд-) ниво поддържането на толкова детайлна информация за текущото натоварване на всеки възел би предизвикало потенциално неприемлив свръхтовар от информационни съобщения към съответния централизиран или разпределен монитрингов процес. Освен това подобна прецизност е и ненужна, тъй като истинските цели на балансирането – ефективността и бързодействието – се постигат не за сметка на буквално изравняване на натоварването на възлите, а просто чрез недопускане на възли с нулево натоварване във всеки момент на обслужването. За целта е достатъчно да знаем дали има възли с нулево натоварване и също дали има възли, които се доближават до такова състояние – т.е. достатъчно е да разграничаваме състоянията

- ***донор*** – претоварен възел, който е кандидат за потенциален балансиращ трансфер към други възли;

[1] в мрежа с общодостъпно предаване логическият граф на обмен е всеки-към-всеки и един-към-всички.

- ***приемник*** – възел, който е желателно да бъде натоварен с нови задания, за да не изпадне инцидентно в състояние на нулево натоварване.

Тези две състояния очевидна са минималният праг на точност на информацията за системата. В сървърните операционни системи той се измерва с брой задания за обслужване в локалната системна опашка[1] и предвид на казаното в предходния параграф е желателно този праг да бъде стойност, достатъчно близка до нулата, но все пак съобразена със средната грануларност на обслужваните задания, т.е. средното време за обслужване на задание, съпоставено със средното време на цикъла на балансиращ трансфер на задания. Логично е да се очаква, че при едра грануларност прагът между състояние донор и приемник е по-нисък, т.е. по-близък до нулата, отколкото при системи за обслужване на заявки с по-фина грануларност. Понятието грануларност ще бъде въведено на базата на параметрите на моделите в гл. 3.

По-висок праг на точност предполага въвеждане на повече състояния на възлите – например между донор и приемник може да се въведе междинно състояние – да го означим неутрално състояние – при което възелът нито приема допълнителни задания от донори, нито пък изпраща задания към приемници, като по този начин се редуцира системния свръхтовар. Можем да продължим с увеличаване на броя състоянията на възлите в системата – въвеждайки например състояние супер-приемник – за възли, които не само, че са приемници, но и са с нулева или близка до нулата дължина на локалната системна опашка от процеси за обслужване. Оптималният брой и стойност на праговете между състоянията на възлите може да бъде оценен в резултат на моделиране или експериментално. Общият принцип е, че неголемият брой прагове се избира със стойности, които са близки до нула. Нека напомним тук една находчива схема на такива прагове, предложена в [15], при която системата от прагове се разполага експоненциално

[1] моментното процентно натоварване на ядрата не е достатъчно добър параметър, тъй като то отразява освен локалното натоварване на възела, така и локалното планиране и също характеристики на текущите заявки за обслужване – примерно заявки, които изчакват синхронизиращ обмен, може да се отразят в ниски процентни стойности на локалното натоварване, но реално възелът да не е с по-малко задания за обслужване от другите възли в клауда.

по броя задания за локално обслужване (0, 1, 2, 4...) – рефлектирайки по този начин намаляващата ценност на информацията за конкретната дължина на локалната системна опашка с нарастване на броя на заявки в обслужване.

Друга ска́ла в общия модел на СВ е ***инициативата за трансфер***, която е всъщност и протокола на трансферната услуга. Инициативата за трансфер може да бъде:

- инициатива на донора;
- инициатива на приемника;
- симетрична инициатива.

В първия случай – при ***инициатива на донора*** – балансиращият трансфер на заявки за обслужване – т.е. самата трансферна услуга – се изпълнява от донора на задания, т.е. от възела с по-голям товар към възела без или с по-малък товар. При инициатива на приемника логиката се обръща – възел, който е с нулево или минимално натоварване (примерно брой заявки за обслужване под даден праг) изпраща служебно съобщение към възел-донор за прехвърляне на обслужвани заявки. При симетричната инициатива и донорите и приемниците стартират прехвърляне задания в съответната посока. Априори не би могло да се оцени предимството на всеки от трите подхода. Един анализ в [4] доказва, че трансферната инициатива има значително влияние върху общата ефективност на балансиращата схема по отношение скорост на обслужване и генериран свръхтовар. Краткият извод от цитирания анализ е че при ниско и средно натоварване на разпределената система – примерно до 75-80% от процесорното време на всеки възел – инициативата на донора дава по-добри резултати за средно време за изпълнение на отделните заявки. При високо натоварване обаче свръхтовара от трансферния протокол, изпълняван от възлите донори, води до бърза деградация на производителността. На какво обаче се дължи по-добрата производителност с инициатива на донора при ниско и средно натоварване?

Тук има една особеност, която прави донорната иницитива на практика основния принцип на управление на балансирането. При иницитива на донора (и прилагане на

управление по събитие, което значително преобладава над управлението по време) инцидентът на претоварване на възела, т.е. преминаването му от състояние приемник (или въобще „пред-донорно”) към донор, възниква с постъпването на една или повече нови заявки за обслужване. Следователно обект на инициираните в този момент балансиращи процедури и планиране са именно тези новопостъпили заявки, които се пренасочват към възел в състояние приемник. Това съвпадение на двете събития – постъпване на необслужени заявки и изпълнение на балансиращия контролен цикъл – е благоприятно, тъй като не е свързано с допълнително изчакване (освен ако не попаднем в неблагоприятната ситуация да няма нито един възел-приемник за момента, но тогава няма и смисъл да се планира и изпълнява балансиращ трансфер).

За контраст при ***инициатива на приемника***, както и при симетрична инцитива, инцидентът на изпадане на възел в състояние приемник, или дори в нулево състояние (което е най-нежелателно), настъпва в момента на напускане на една или повече изпълнени заявки, който очевидно не е непременно синхронен с момента на постъпване на нови заявки. Следователно за да натоварим такъв възел-приемник, или трябва все пак да изчакаме постъпването на нова заявка – което е напълно асинхронно събитие, или да прехвърлим на него изпълнението на вече частично изпълнена заявка от възел-донор, което – ако изобщо е възможно – очевидно навлича допълнителен и то контекстно-зависим свръхтовар – ако искаме да съхраним статуса на изпълнението, а не да започнем обслужването отначало. При това положение изобщо е чудно защо все пак инициатива на приемника може да даде по-добра производителност в системи, близки до пълно натоварване – 80-95%. Едно интуитивно, но все пак правдоподобно обяснение е, че в претоварените системи почти няма вероятност за възли-приемници и следователно балансиращият цикъл заедно с целият негов свръхтовар затихва. И обратното – при инициатива на донора точно в претоварено състояние с висока честота, но с малка вероятност за успех, започват да се търсят възли-приемници, което генерира допълнителен свръхтовар.

Очевидно дилемата между двете схеми на инициативата за трансфер може да се реши чрез адаптивна иницитива. Адаптивната иницитива – за разлика от симетричната! – означава изпълнение само на единия принцип в зависимост от моментното общо натоварване на системата. При симетричната иницитива моментите на търсене на комплементарен възел (с който да бъде извършен балансиращия обмен на задание) настъпват и при напускане и при постъпване на нови задания. Независимо, че някои моделни резултати дават добра производителност при симетрична иницитива [4], практичността на схеми за СВ, включващи иницитива на приемника, остава съмнителна поради посочения проблем с асинхронните събития на постъпване на нови заявки спрямо моментите на взимане на балансиращи решения и поради проблемния трансфер на частично изпълнени задания.

В разгледаните модели (гл. 3. и 4.) ние представяме резултати, отнасящи се само до иницитива на донора, макар че моделите могат да бъдат преформулирани и анализирани и за инициатива на приемника. Считаме, че такъв анализ по отношение на инициативата би препотвърдил поне в някаква степен цитираните ранни резултати на Eager-Lazovska-Zahorian, но едва ли би имал някаква практична стойност, тъй като по много косвени признаци може да се прецени, че принципът на иницитива на приемника н е с е п р и л а г а от никоя операционна система[1].

Както виждаме моделното пространство на проблема за балансиране на изчислителния товар е многомерно, което предполага не-малка сложност на моделите ако целта е съпоставимост и адекватност на моделните резултати.

[1] само́ по себеси това твърдение няма как да бъде доказано с позоваване на конкретни източници, тъй като принципите на ресурсното планиране – както локално, така и клъстерно – не се оповестяват или са описани неясно и неконкретно в съответните фирмени документи (очевидно бидейки възприемани като важно know-how).

2.2.2. Параметризация на модела

Нека представим параметрите, които са необходими за квантифициране на така описаните балансиращи стратегии.

Входните параметри на модела описват ресурсите на клауд-системата, характеристиките на потока от заявки за обслужване, избраният модел на балансиране – включително логическата топология.

На първо място това е ***размерът на системата***, оценяван чрез

- n – брой еднотипни възли в клауд-клъстера и
- c – брой процесорни ядра във всеки възел.

Намираме за уместно да детайлизираме модела с въвеждане на броя локални обслужващи ядра, тъй като този параметър влияе директно на минималния праг на локално натоварване в брой обслужвани заявки, а както отбелязахме при съвременните сървърни архитектури броят процесорни ядра обикновено е висок и почти не се среща клауд-инфраструктура, в която c=1.

Всъщност ограничението за еднотипност на възлите лесно може да бъде преодоляно ако представим системата като състояща се от няколко класа еднотипни възли и извършим анализа по представения в т. 3. метод за всеки от класовете възли, като накрая обобщим получените резултати за цялата система. В тази работа сме представили анализа на модели на хомогенна система. Тъй като клауд-клъстерите обикновено се проектират и изграждат като специализирана сервизна инфраструктура, в повечето случаи те са именно хомогенна система. Възможните изключения от хомогенната клъстерна архитектура са по-скоро във функционална специализация на някои възли – най-често например комуникационни сървери в клъстера. Приемаме, че такива специализирани сървери се изключват от обхвата на моделиране, още повече, че те обикновено изпълняват съответните системни функции и не обслужват потребителски заявки към клауд-клъстера.

Процесът на *постъпване на заявки* за обслужване се характеризира с параметрите:

- Λ – темп на постъпване на заявки в клъстера – накратко глобален темп на заявките и
- λ – темп на заявките във всеки възел – локален темп.

Друга характеристика на процеса на постъпване на заявки е моделът на тяхното ***начално разпределяне между възлите***:

- централизиран модел на постъпване – да го означим CS; при него клиентските заявките постъпват в един специализиран клъстерен контролер, който ги разпределя между изпълнителните сървери в клъстера (това е най-разпространения модел на обслужване в клауд);
- глобално разпределено постъпване – да го означим p2p; потребителските заявки постъпват във всички възли, а те ги преразпределят балансиращо помежду си – включително по Браунова (към всеки друг възел) или дифузионна (само към „съседните" им възли) схема;
- локално разпределено постъпване – да го означим Lp2p; потребителските заявки постъпват в няколко интерфейсни възли, а те ги преразпределят балансиращо между останалите обарботващи възли – отново по Браунова или дифузионна схема.
- па повод на схемите и обхвата на действие на CB-процедурите, които означаваме като Браунови и дифузионни, е необходим параметърът ***локалност***, който представлява броя „съседи" на всеки от балансиращите и обработващи възли – ν, както и тяхната топология, представена от матрицата на съседство.

Характеристиката на ***процеса на обслужване*** на заявките, т.е. на тяхното напускане на системата, е стохастичен прицес с локален темп на обслужване от всяко ядро ω.

Прагът на точност на наблюдението се представя от броя състояния на натоварване, в които може да се намира даден възел. В съответствие с казаното в 2.2.1.

ние се ограничаваме с три (а не повече) състояния на възлите: донор (да означим това състояние със символа $\mathcal{S}$), приемник ($\mathcal{R}$) и неутрален ($\mathcal{N}$). За такъв праг на точност са ни необходими два числови прага:

- T_1 е горният праг на състоянието $\mathcal{R}$ – т.е. максималния брой на заявки, обслужвани от възлите с ниско натоварване;
- T_2 е долният праг на състоянието $\mathcal{S}$ – т.е. минималният брой на заявки, обслужвани от възлите с високо натоварване.

За да се определи текущото състояние на даден възел, дължината на неговата локална системна опашка от задания l се сравнява с двата прага. Не всяка промяна на l, а само промяна на състоянието на възела ($\mathcal{R} \leftrightarrow \mathcal{N} \leftrightarrow \mathcal{S}$) е събитие, което се регистрира от услугата за наблюдение.

Тук ще покажем, че ***управлението на преходите*** между установените състояния също може да бъде предмет на диверсификация. Обичайно и традиционно тези прагове и броя процесори/ядра във всеки възел се намират в следното съотношение $c \leq T_1 \leq T_2$. Така е във всички цитирани източници по темата СВ. В едно ранно наше изследване от 1997 г. предлагаме и обосноваваме ефективността на едно инвертирано съотношение на праговете: $T_1 > T_2$, което означаваме като ***инверсия***. При традиционното безинверсно съотношение на двата прага последователните преходи между крайните състояния на възлите ($\mathcal{R}$ и $\mathcal{S}$) преминават през неутралното междинно състояние ($\mathcal{N}$), при което възелът не извършва нито балансиращо изпращане нито балансиращо приемане на задания, намалявайки по този начин общия системен свръхтовар. При инверсия обаче логиката на междинното състояние се променя – възелът едновременно е и донор и приемник на задания – означаваме го като $\mathcal{RS}$ (“re-sender”) – ***транзитен***. Очевидно такава прагова схема стимулира балансиращото пре-предаване на задания и може експериментален или моделен анализ да се окаже, че тя е ефективна при определени модели на натоварване на системата и по-специално когато имаме дифузионно разпределение на балансирането

с ниски стойности на локалността $v<<n$. Разбира се, в този случай балансиращите протоколи не трябва да допускат обмен на $\mathcal{RS}$-възел към друг $\mathcal{RS}$-възел за да не се създава безполезен системен обмен. Възел в състояние $\mathcal{RS}$ може да изпраща новопостъпило задание само към $\mathcal{R}$-съсед (ако има); $\mathcal{S}$-възел предава задание към $\mathcal{RS}$-съсед само в случай, че няма $\mathcal{R}$-съсед. Това е праговият хистерезис при три различими товарни състояния на възлите в клъстера – съответно разграничени с двата прага T_1 и T_2.

Можем да преформулираме прагова инверсия и тогава, когато различимите състояния са само две – $\mathcal{R}$ (приемик) и $\mathcal{S}$ (донор), съответно с един праг между тях – T. В този случай са ни необходими две стойности за T: $T_{\mathcal{R}}$ и $T_{\mathcal{S}}$ като $T_{\mathcal{R}} > T_{\mathcal{S}} > c$ (при обратно неравенство между $T_{\mathcal{R}}$ и $T_{\mathcal{S}}$ би се получил вариантът с 3 състояния). Това състояние наричаме ***хистерезис***. Логиката на двойния хистерезисен праг тук е следната: с постъпването на задания за локално обслужване в дадения възел преходът му от приемник към донор става при по-високо натварване $T_{\mathcal{R}}$, отколкото обратният преход от донор към приемник $T_{\mathcal{S}}$. Образно казано възелът „не бърза” да си смени състоянието и в двете посоки с цел да се минимизира системния свръхтовар на балансирането. Както отбелязахме, този похват не е описван и изследван в други източници и в т. 4. ние ще предложим само моделна оценка на неговото въздействие върху производителността на клауд-системата. За съжаление нямаме експериментални имплементации, които да демонстрират евентуалната полза от прагов хистерезис.

Междинните или ***вътрешни параметри*** са тези, които се получават от входните параметри на базата на анализ на модела и са необходими за намирането на изходните параметри, но нямат непосредствена интерпретация при оценка на производителността на системата и следователно не могат да се разглеждат като „резултат” от моделирането. Все пак важно е те да се обявят, тъй като резултатите, които се получават за тях, служат за индикатор на коректността на направените анализи и се явяват косвено потвърждение на крайните резултати. В нашите модели оценяваме следните междинни параметри:

- локален темп на балансиращо постъпване на задания от донори ρ;
- локален темп на балансиращо изпращане на задания към приемници σ;
- ефективен темп на постъпване на задания в даден възел λ_ε;
- ефективен темп на напускане на обслужени или прехвърлени задания μ_ε;
- локален коефициент на натоварване ϕ;
- локално натоварване – среден брой задания за обслужване във възела l;
- коефициент на глобалното натоварване – отношението между темповете на постъпване и напускане на задания от системата – γ[1];
- среден брой възли в дадено състояние: донор, приемник, неутрален или транзитен – съответно N_S, N_R, N_N и N_{RS};
- вероятността даден възел да се намира в едно от възможните състояния – в нашите модели те са две или три – P_S, P_R, P_N и P_{RS}.

Ако системата е хетерогенна и се състои от няколко класа възли (класът обединява възлите с еднакви ресурси), всички така изброени параметри, които се отнасят до отделни възли (напр. вероятността даден възел да е донор – P_S) трябва да бъдат допълнително индексирани с номера на класа. В този случай анализът се извършва по отделно за класовете и след това се обобщава за системата. Един по-характерен такъв пример е клауд-клъстер с един комуникационен и контролен сървер, в който постъпват всички задания и той ги разпределя балансиращо между останалите еднотипни изпълнителни сървери в клъстера. В този случай например бихме имали CS-система с два класа:

[1] от гледна точка на моделите на реални обслужващи процеси в клауд параметърът коефициент на глобалното натоварване γ е м е ж д и н е н, но всъщност при анализа и съпоставката на група от абстрактни процеси на обслужване е удобно той да се използва като входящ; по този начин можем да изследваме производителността на дадена система в условиятана различно относително натоварване – т.е. да променяме γ като в х о д я щ параметър, а да преизчисляваме като вътрешни някой/и от входните параметри и да получим съответното семейство криви – както е демонстрирано в т. 3.

- клас 1: един контролер с голям темп на постъпване, но и на напускане на заданията и
- клас 2: n-1 изпълнителни сървери, „съседни” на контролния, а също „съседни” с [част от][1] останалите сървери в клас 2; те са с по-ниски темпове на постъпване и на напускане на задания.

Друг моделен подход за същата система е да се абстрахираме от производителността на контролния сървер и да разглеждаме системата като хомогенна [L]p2p система, опростявайки по този начин анализа й (но разбира се ще загубим възможността да анализираме параметрите на обслужване на контролния сървер).

Да изброим очакваните резултати т.е. ***изходните параметри*** на модела.

- средно време на обслужване на заданията в клауд-клъстера T;
- локален (за даден възел) темп на балансиращ обмен на задания θ;
- локален темп на генериране на съобщения на услугата за наблюдение δ;
- глобален (за клауд-клъстера) темп на системните съобщенията χ (т.е. тези, които формират целия системен свръхтовар – обмена на задания и обмена с цел наблюдение на системата);
- темпове на локалния и глобалния процес на напускане на изпълнените задания – съответно μ и M.

Ще се ограничим с изброяването на тези резултати, като непосредствен продукт от моделирането на системната производителност, но на тяхна база могат да се получат и други параметри – например:

- средно време на престой на възлите (включващо времето през което поне едно процесорно ядро във възела е в престой, с други думи времето, за което $l<c$);

[1] в зависимост от избраната схема p2p (v = n-2) или Lp2p (v < n-2).

- оптимален брой обслужващи възли при зададен глобален темп и грануларност на заданията – $n_{\max}$ със запазена линейност на ускорението на разпределената обработка.

Моделирането и анализът на СВ имат за цел по-ефективно обслужване на клауд-инфраструктурата. От потребителска гледна точка това обикновено означава по-кратко време на обслужване. От гледна точка на доставчика на услуги в клауд (включително доставчика на инфраструктурата като услуга) ефективността се оценява както по времето за обслужване, така и по коефициента на използване на ресурсите. От гледна точка на обществото като цяло ефективността може да се оценява и чрез непарична цена на обслужването – както е при „зелените" клауд-системи. Както отбелязахме в 1.2. при многофакторен анализ на клауд ще не и необходим векторен модел на ресурсите и натоварването.

3. Модели за управление на обслужването

Описаните в предходната глава параметри на обслужването в клауд освен като входни, междинни и крайни (т.е. резултати), можем да групираме и в друга ска́ла: ресурсни, товарни и контролни – с очевидна семантика.

Контролните параметри характеризират избраните стратегии за управление на качеството на обслужване, което – особено в публичен клауд за информационно обслужване на множество потребители – се изразява предимно в бързодействие на услугите при конкурентна цена т.е. при висока ефективност на разгърнатите ресурси. Нека отделим контролните параметри, които описват управление на бързодействието и ефективността, за да съставим моделното пространство, което ще анализираме. Те са:

- контролна схема: централизирана (CS), разпределена (p2p, „Браунова”) или локална (Lp2p, „дифузионна”);
    - локалност ν (или логическа топология – при локалните схеми);
- праг на точност:
    - брой състояния на възлите;
    - стойност на праговете между състоянията;
- инициатива на балансирането (т.е. събитие, стартиращо балансиращия трансфер на задания):
    - на донорите;
    - на приемниците;
    - симетрична;
    - хистерезисна.

Възможно е да бъдат добавени и други параметри в този списък, но считаме, че именно така изброените контролни параметри имат най-голямо значение за качеството на обслужване в клауд. Броят им, както и този на възможните им стойности – макар и да не е голям – задава все пак едно значително моделно пространство. Тук ще се спрем на няколко по-характерни случая на модели от това пространство, подбрани или поради тяхната разпространеност в реалните клауд-инфраструктури, или – точно обратно – тъй като няма данни да са прилагани или изследвани където и да било (това се отнася за Брауновите, дифузионните[1] и хистерезисните схеми).

Съгласно параметризацията от гл. 2. разглеждаме хомогенен клауд-клъстер, състоящ се от n обслужващи възела, всеки с c процесорни ядра и свързани във високоскоростна мрежа с общодостъпно предаване.

Процесите на постъпване и на обслужване на задания са независими и асинхронни, поради което можем да ги апроксимираме като Пуасонови стохастични процеси със средни стойности:

- темп на обслужване на задания от всяко процесорно ядро ω;
- темп на постъпване на задания във всеки възел $\lambda_{\mathcal{R}|N|S|\ldots}$; индексирането на локалния темп на постъпване на задания всъщност би трябвало да съответства на числовия индекс на възела, но тук сме го свели до индекс на състоянието на възела, тъй като системата е хомогенна и трябва да приемем, че възлите в еднакво състояние имат и еднакво натоварване.

Описаната в гл. 2. както и тук праметризация на модела може непосредствено да се представи с апарата на теорията на масовото обслужване на Клайнрок – Ерланг. Основен метод за анализ на такива модели е стимулационното моделиране. Този метод сме приложили в [8, 9]. Симулационните методи дават бързи резултати и в повечето случаи евентуалните ограничения по отношение скалируемост на моделите са

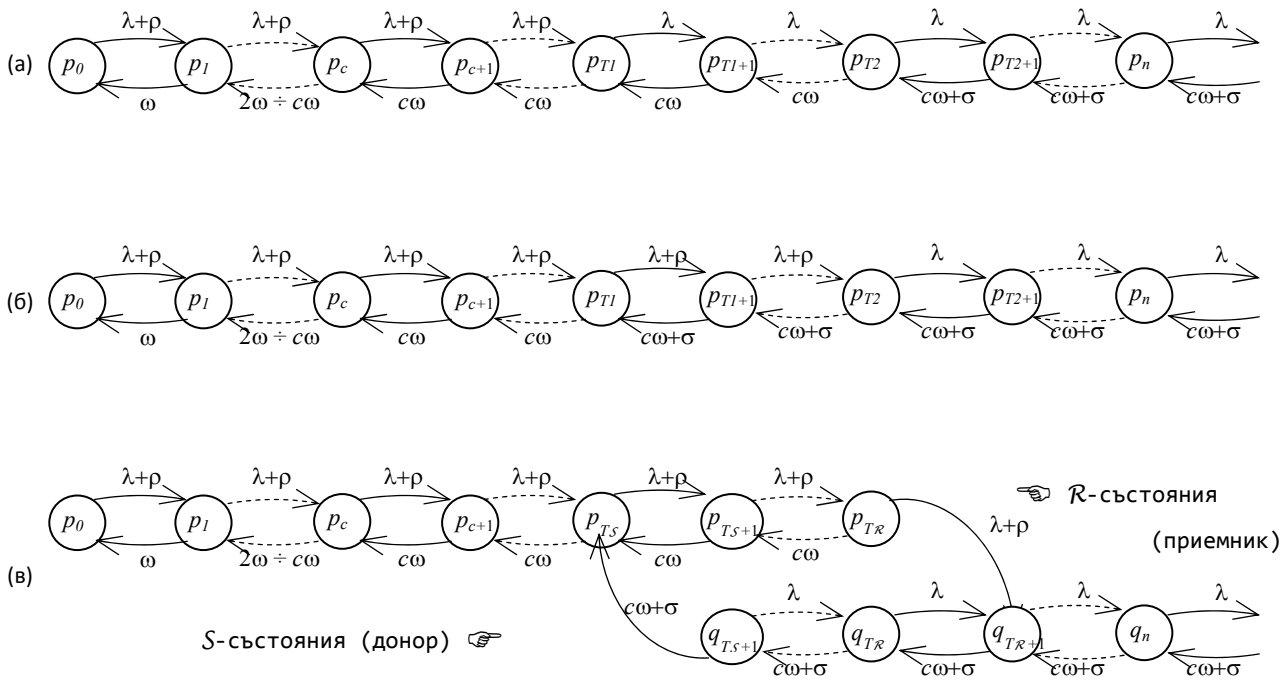
[1] дифузионни схеми са прилагани в мрежи без общодостъпно предаване, но не и в мрежи с такова – каквито са мрежите с конкурентен достъп до общ комуникационен канал.

преодолими. Най-важните проблеми при тях са коректността на представяне на абстрактния модел със средствата на езика на компютърния симулатор, както и коректността на работа на самата система за симулации. Минимални скрити грешки в стимулационното приложение или в стимулационната система могат да доведат некоректни резултати. Ако симулационните резултати са недостоверни (например по-кратко време за обслужване в система с по-високо натоварване), тогава тяхната некоректност е лесно установима. Но ако те изглеждат достоверни, коректността им все пак остава недоказуема априори. Да посочим пример от собствен опит. Представените по-долу модели сме оценявали и с помощта на стимулационната система GPSS Minuteman и получените резултати изглеждаха напълно достоверни до момента, в който в режим debug и постъпково изпълнение на симулацията не установихме, че дължината на системните опашки съответства на цикличен избор на опашка, вместо на заявения в стимулационната програма случаен избор. Разбира се, този дефект на системата е отстраним чрез вграждане на съответна функция (както постъпихме с моделите в цитираните публикации), обаче въпросът с надеждността на симулационните резултати остава.

3.1. Диаграми на състоянията и преходите на възлите

Основният аналитичен подход за решаване на модели в теорията на масовото обслужване в съответствие с традицията на Клайнрок-Ерланг са веригите на Марков.

При такъв анализ възможните състояния на всеки възел от клауд-клъстера се представят чрез текущия брой обслужвани в тях задания l и когато l премине през единия, двата (или в общия случай няколкото) прага $T_{1,2...}$ се променя и състоянието на възела – $\mathcal{R}$, $\mathcal{N}$, $\mathcal{S}$ и т.н. Маркировката на всяко състояние е вероятността възелът да се намира в това състояние – или с други думи относителното време, през което възелът се намира в това състояние от общото време на обслужването – p_i.



Фигура 3.1. Диаграми на преходите и състоянията на възел с: ***(а)*** дифузионно или Брауново балансиране и три състояния; ***(б)*** дифузионно или Брауново балансиране, три състояния и транзит; ***(в)*** дифузионно или Брауново балансиране, две състояния и хестерезис.

Преходите между състоянията на възела възникват с темп, който е продукт от действието на няколко стохастични процеса, характеризирани чрез входящите и вътрешните параметри λ, ω, ρ и σ (с точност до индексите). Съответните темпове са маркировката на преходите. ***Фигура 3.1.(а)*** представя диаграма на състоянията и перходите на произволен възел от клауд-клъстера. Процесът на постъпване на задания във възела е с темп λ, а на обслужване (от всяко процесорно ядро на възела) – с темп ω. Към тези два процеса са добавят и процес на балансиращо приемане на задания от претоварени възли-донори с темп ρ – когато възелът е приемник ($\mathcal{R}$) и процес на изпращане на задания към възли-приемници с темп σ – когато възелът е донор ($\mathcal{S}$). Информационните състояния – така, както се представени на фигурата – са три: с междинно неутрално състояние и е в сила съотношението $c < T_1 < T_2$ – т.е. съотношението на праговете е ***без хистерезис/транзит***. Затова в състоянията между T_1 и T_2 няма балансиращ обмен (ρ и σ) с други възли в клъстера. С тази диаграма можем да

представим и случаят на балансиране в система само с две информационни състояния и един праг: тогава $T_1 = T_2$. Параметърът локалност v, който разграничава Брауновите (v=n) от дифузионните (v<<n) схеми, не е елемент на тази диаграма, и следователно тя е обща за тези две схеми на баланисране, но той участва имплицитно при изчисляване на параметрите ρ и σ; следователно анализите на двата случая ще се извършат самостоятелно. Така че p2p- и Lp2p-схемите са представени на тази диаграма. Частен случай от нея е СВ с две товарни състояния на възела (и един праг T) – достатъчно е да установим $T_1 = T_2$.

Същата диаграма представя и централизираното балансиране CS по следния начин. Централният балансиращ процес разделя входящия поток задания между всички възли-приемници в клъстера, ако няма такива – между всички неутрални възли, и ако такива съща няма – тогава между всички останали – които (уви) са в състояние донор. Следователно диаграмата е същата, но анализа й ще се различава, тъй като зависимостите на преходните темпове λ, ω, ρ и σ от сродения брой възли в дадено състояние (донор, приемник, неутрален или транзитен – съответно $N_{\mathcal{S}}$, $N_{\mathcal{R}}$, $N_{\mathcal{N}}$ и $N_{\mathcal{RS}}$) ще бъдат различни.

За представяне на схемите ***с инверсен транзит*** е достатъчно в диаграмата *3.1. (а)* да поставим праговите стойности в обратно съотношение ($c < T_2 < T_1$). За да имаме по-голяма яснота във фазата на анализа тук сме представили вариант на тази диаграма, при който праговете са подредени според традиционното очакване $T_2 > T_1$, но действието на балансиращите системни процеси е съобразено с хистерезисна схема. Тъй като схемата е с два прага и три състояния, както отбелязахме в т. 2.2.2., междинното състояние е $\mathcal{RS}$ (“re-sender”, транзитен) – ***фиг. 3.1.(б)***.

Нека за сравнение да разгледаме случаят на СВ ***с инверсен хистерезис***, при само две състояния на възела – донор $\mathcal{S}$ и приемник $\mathcal{R}$ – ***фиг. 3.1.(в)***. В този случай състоянието на възела – дали ще изпраща или ще приема балансиращи натоварването задания – зависи не само от текущия брой задания за изпълнение в локалната му опашка, но и от предишното му състояние. Затова само индексът на вероятностите p_i не е достатъчен да

разграничи възможните състояния и съответните вероятности на $\mathcal{S}$-възел са означени с друг символ на вероятността – q_i.

3.2. Случаи на балансиращите модели

Както може да се прецени от представените диаграми на преходите и състоянията на възлите, апаратът на веригите на Марков дава широки възможности за формулиране на различни балансиращи схеми. Тук разглеждаме схеми само с две до три състояния на локалното натоварване, но очевидно подобни диаграми могат да се съставят и за повече състояния. Също така е възможно да приложим схема с хистерезис (вместо транзит на фиг. 3.1.(б)) за три състояния на възела. Тогава хистерезисът по модела на диаграма (в) може да се установи при прехода $\mathcal{R}$-$\mathcal{RS}$, но и при $\mathcal{RS}$-$\mathcal{S}$, а също и при двата прехода. Очевидно вариантите дори и само при два-три прага на натоварването са множество.

Да не забравяме, че в общия случай клъстерът може и да не се състои от еднотипни възли. Тогава анализът се извършва за всеки клас възли, след което се обобщава за цялата система. В този случай може да се окаже, че различните класове възли дават оптимална производителност при различаващи се СВ-схеми.

Нека за конкретност на изложението да се спрем на няколко характерни случая на така описания модел на СВ, чийто анализ би ни дал представа за въздействието на различните балансиращи схеми върху общата производителност на клауд-клъстера.

Разглеждаме хомогенен клауд-клъстер; нека опишем най-напред ресурсите му. Клъстерът се състои от n обслужващи възела, всеки с c паралелно работещи ядра, свързани във високоскоростна мрежа с общодостъпно предаване (broadcasting). Обслужването на отделните асинхронни независими задания е стохастичен Пуасонов процес със среден темп за всяко процесорно ядро ω.

Натоварването на клъстера е централизиран стохастичен процес от задания за обслужване, които се разпределят между възлите. Централизираният разпределящ

процес обикновено не поддържа детайлна моментна информация за текущото локално натоварване на възлите, въз основа на която да взима планиращи решения [(SwiftWater. 2010)]. Това е така поради:

- съображения за скалируемост на клъстера;
- съпоставимост на времето за изпращане на евентуални информационни съобщения от обслужващите възли и времето за локално обслужване на заданията с фина и средна грануларност, от което следва:
    - непрактично висок системен свръхтовар и[ли]
    - недостатъчна рефлективност на централизираната таблица за наблюдение на локалното натоварване;
- принципната непредвидимост на времето за обслужване на всяко задание [12].

Така че на практика един централизиран балансиращ сървер обикновено извършва някаква форма на случайно равномерно разпределяне на заданията между възлите в клъстера – например циклично или пък (което е най-разпространено) – чрез разпределени хеш-таблици (DHT) [18]. По-подробно този въпрос е разгледан също в [24, 25]. Затова в нашите моделни случаи процесът на постъпване на задания за обслужване в клъстера е също Пуасонов с темп $\Lambda=n\lambda$, който се разпределя с равна вероятност между всички възли на Пуасонови под-процеси с параметър λ.

Случай $\mathbb{DF}$ (дифузионен – Lp2p). Всеки възел има:

- три различими състояния – донор, неутрален и приемник, разграничени от праговете на дължина на локалната си опашка от задания T_1 и T_2 ($c \leq T_1 < T_2$);
- v ($v << n$) съседи, които наблюдава и информира за всяка локална промяна на своето състояние, т.е. за преход от донор към неутрален и т.н.

Когато възелът е донор, той изпраща по възможност всяко новопостъпило задание към приемник между своите съседи. Инициативата е на донора: момент за балансиращ обмен е само постъпването на ново задание; ако то не бъде изпратено в този момент към съсед-приемник поради липса на такъв, то остава за локална обработка и повече не подлежи на обмен в следващ момент, тъй като евентуално вече ще е частично обслужено съгласно локалната планираща процедура. DF се представи с диаграмата на фиг. 3.1.(а).

Случай BR (Браунов, p2p). Като DF, но $v = n$, т.е. всички възли в клъстера са съседи, между които донорът търси евентуален приемник. BR също съответства на диаграмата на фиг. 3.1.(а).

Случай DT (дифузионен с транзит). Като DF, но $c \leq T_2 < T_1$, т.е. междинното състояние на възела между приемник и донор и транзитно с пре-предаване на задания. DT съответства на диаграмата на фиг. 3.1.(б).

В следващата глава ще представим числовия анализ на така описаните модели.

4. Анализ на моделите за управление

Анализът на случаите DF, BR и DT (както и на други СВ случаи) се свежда до решаване на системата от равновесни уравнения на техните диаграми когато са изпълнени условията за ергодичност. Тъй като графите на тези Марковски вериги са свързани, необходимото и достатъчно условие за ергодичност на системата е относителното натоварване да е под 100%. По-нататък в т. 4.1. ще формулираме прецизно тази стойност, както и останалите параметрични зависимости.

4.1. Равновесни уравнения на моделите на възел

Анализирайки DF и неговата диаграма на фиг. 3.1.(а), стигаме до следната система от полиномиални уравнения спрямо равновесните стойности на вероятностите p_i, които съответстват на относителното време, през което всеки обслужващ възел има i на брой задания в локалната си системна опашка:

$$(\lambda+\rho)p_i = (i+1)\omega p_{i+1}, \qquad i \in [0, c] \tag{e1}$$

$$(\lambda+\rho)p_i = c\omega p_{i+1}, \qquad i \in [c+1, T_1\text{-}1] \tag{e2}$$

$$\lambda p_i = c\omega p_{i+1}, \qquad i \in [T_1, T_2\text{-}1] \tag{e3}$$

$$\lambda p_i = (c\omega+\sigma)p_{i+1}, \qquad i \in [T_2, \infty) \tag{e4}$$

$$\sum_{i=0}^{\infty} p_i = 1, \tag{e5}$$

$$\rho\, P_{\mathcal{R}} = \sigma\, P_S, \qquad \text{където} \tag{e6}$$

$$P_{\mathcal{R}} = \sum_{i=0}^{T_1-1} p_i, \tag{e7}$$

$$P_{\mathcal{S}} = \sum_{i=T2+1}^{\infty} p_i\,, \tag{e8}$$

$$\sigma = \lambda\, P_{\Sigma_{\mathcal{R}}}, \qquad \text{където} \tag{e9}$$

$$P_{\Sigma_{\mathcal{R}}} = 1\text{-}(1\text{-}P_{\mathcal{R}})^{v}. \tag{e10}$$

Тази система е съставена по следния начин. Уравнения (e1) ÷ (e4) представляват равновесните уравнения на диаграмата на $\mathbb{DF}$. Те са точно i-1 на брой спрямо i на брой неизвестни ($i \in [0, \infty)$). Следователно за пълнота на системата ни е необходимо още едно уравнение и както обикновено то изразява закона за пълната вероятност – (e5). С това системата уравнения би била пълна (и линейна) ако параметрите ρ и σ бяха независими и известни. Те обаче не са входни, а междинни параметри (т. 2.2.2.) и са продукт на съвместната работа на всички обслужващи възли, поради което е необходимо да ги дефинираме, допълвайки системата с двете уравнения (e6) и (e9). Останалите параметри в системата уравнения са входни параметри.

По-специално (e6) отразява факта, че при процеса на балансиращ трансфер на задания, те нито се създават (т.е. постъпват в клъстера), нито се прекратяват (т.е. обслужват); следователно общия брой изпратени от донорите задания трябва да е равен на броя получен в приемниците задания в клауда. В (e6) коефициентът n е съкратен от двете страни на уравнението. Уравнения (e7), (e8) са спомагателни уравнения, дешифриращи съкратените записи на агрегирани вероятности $P_{\mathcal{R}|\mathcal{S}}$ в (e6). $P_{\mathcal{R}}$ е вероятността дадения възел да е приемник, съответно $P_{\mathcal{S}}$ – същият да е донор.

Уравнение (e9) изчислява локания темп на балансиращ трансфер към приемници σ, като взема предвид, че при инициитива на донора събитието балансиращо изпращане на задание възниква винаги, когато ново задание постъпи за обслужване с темп λ в $\mathcal{S}$-възел и при това този възел има поне един $\mathcal{R}$-съсед. Вероятността да се изпълни

последното условие според (e10) е $P_{\Sigma_{\mathcal{R}}}$. (e10) се базира на v на брой последователни независими Бернулиеви опита[1] с вероятност за провал на всеки такъв опит $(1\text{-}P_{\mathcal{R}})$.

За до получим системата равновесни уравнения за $\mathbb{BR}$ е достатъчно само да заменим в (e10) v с n – т.е. от Lp2p получаваме p2p. Това е логично да е така, тъй като тази фаза от анализа си извършва на ниво възел (в хомогенна система) и следователно разликата между Lp2p-възел и p2p-възел е само в обхвата на балансиращата схема.

За анализ на транзитния случай DT, както и на хистерезисния случай, представени съответно на фиг. 3.1.(б) и (в), този етап на анализа изисква формулирането по аналогичен начин на съответните системи равновесни уравнения. За DT системата на уравнения е практически във същата форма, както (e1) ÷ (e10), само с корекция на съответните коефициенти. За случаите с хистерезис на фиг. 3.1.(в) равновесната система се извлича по аналогичен начин.

Планът за анализ на параметрите на клауд-клъстера започва с анализ на локалните параметри на даден негов възел (или клас от възли – пре хетерогенност). Това означава да се реши системата (e1) ÷ (e10) спрямо неизвестните p_i. На въпроса за решаване на тази система от уравнения се спираме по-надолу в гл. 5.

4.2. Анализ на параметрите на клъстера

След решаване на локалните стойности на p_i за всеки произволен възел или клас възли в клъстера, ще трябва да получим останалите стойности на междинните и изходните параметри. За целта трябва да изхождаме от конкретния случай на СВ. Тук ще се спрем подробно на анализа на първия случай – $\mathbb{DF}$. По същия начин може да бъде анализиран който и да е от така описаните случаи, както и всеки подобен случай на СВ.

Вече посочихме в 3.2., че

[1] тук има елемент на идеализация т.е. на приближение, понеже се разчита, че броят на инспектираните $\mathcal{R}$-съседи няма да се промени докато трае съответния цикъл на търсене на приемник.

$$\lambda_{\mathcal{R}|\mathcal{N}|\mathcal{S}} = \lambda = \Lambda/n. \qquad (e11)$$

Средният брой задания в даден възел е

$$£ = \sum_{i=T_2+1}^{\infty} ip_i\,, \qquad (e12)$$

а ефективния темп на постъпване на задания за обслужване в него е

$$\lambda_\varepsilon = (\lambda+\rho)P_{\mathcal{R}} + \lambda(1-P_{\mathcal{R}}). \qquad (e13)$$

Ефективния темп на напускане на обслужени или прехвърлени към възли-приемници задания е

$$\mu_\varepsilon = \sum_{i=0}^{c}(i+1)\,\omega p_{i+1} + \sum_{i=c+2}^{T_1} c\omega p_i + c\omega P_{\mathcal{N}} + (c\omega+\sigma)P_{\mathcal{S}}$$, където

$$P_{\mathcal{N}} = 1-P_{\mathcal{R}} - P_{\mathcal{S}}\,. \qquad (e14)$$

Освен това – съгласно теоремата на Little – средното време на обслужване на заданията в този възел е

$$T = £/\lambda_\varepsilon. \qquad (e15)$$

Локалният коефициент на натоварване на за възела (а при хомогенен клъстер – и за цялата система) е

$$\phi = \lambda_\varepsilon / \mu_\varepsilon. \qquad (e16)$$

Средният темп, с който всеки донор изпраща задания към свои съседи-приемници, е

$$\theta = \sigma P_{\mathcal{S}}\,. \qquad (e17)$$

Средният темп, с който всеки възел генерира информационни съобщения (за наблюдение на състоянието) към всички свои съседи, е

$$\delta = (\lambda+\rho)p_{T_1-1} + \lambda p_{T_2} + c\omega p_{T_1} + c\omega p_{T_2}. \qquad (e18)$$

Средният темп на системния комуникационен свръхтовар, съставен от съобщенията за изпращане на задания и информационни съобщение за даден възел, очевидно е

$$\chi = \delta + \theta. \qquad \text{(e19)}$$

Коефициентът (или вероятността) за моментно ненатоварено поне едно ядро за възела е

$$\Theta = \sum_{i=0}^{c} \left(1 - \frac{i}{c}\right) p_i \,. \qquad \text{(e20)}$$

Средният локален (за възела) темп на напускане на обслужени задания е

$$\mu = \sum_{i=1}^{c} i\,\omega p_i + c\omega P_{\mathcal{R}} \,, \qquad \text{(e21)}$$

а средният глобален (за клъстера) темп на обслужване е

$$\mathrm{M} = n\mu. \qquad \text{(e22)}$$

Следователно глобалният коефициент на натоварването на клъстера е

$$\Phi = \Lambda/\mathrm{M}. \qquad \text{(e23)}$$

За хомогенна система би трябвало да получим $\phi=\Phi$ от (e16) и (e23), а за хетерогенна система Φ може да се изчисли и през коефициентите на отделните класове възли ϕ_k, и по съвпадането на така получените две стойности за Φ може да се съди за коректността на направените изчисления. Впрочем при удачен избор на балансираща схема между отделните ϕ_k не би трябвало да има голяма разлика – те трябва да са максимално близки по стойност.

5. Числов анализ на моделите

Да се върнем към въпроса с решаването на системата полиномиални уравнения (e1) ÷ (e10).

Най-непосредствената възможност за това е използването на съответни компютърни приложения за математически изчисления. За съжаление нашият личен опит със системата Wolfram Matematika не беше окуражителен, но по никакъв начин не настояваме, че това мнение е меродавно.

Формулирането на тези несложни уравнения на съответния входящ език не представлява проблем и системата генерира алгебричен вид на зададените уравнения. В следващите редове привеждаме за илюстрация фрагмент от такава генерация.

$$\left(\begin{array}{l}
p[0]\left(15.625+\dfrac{15.625\left(1-(1-p[0]-p[1]-p[2])^4\right)\left(p[6]+\dfrac{p[6]}{1-\dfrac{15.625}{8.68056\,c+15.625\left(1-(1-p[0]-p[1]-p[2])^4\right)}}+p[7]+p[8]+p[9]+p[10]\right)}{p[0]+p[1]+p[2]}\right)==8.68056\,p[1]\\
p[1]\left(15.625+\dfrac{15.625\left(1-(1-p[0]-p[1]-p[2])^4\right)\left(p[6]+\dfrac{p[6]}{1-\dfrac{15.625}{8.68056\,c+15.625\left(1-(1-p[0]-p[1]-p[2])^4\right)}}+p[7]+p[8]+p[9]+p[10]\right)}{p[0]+p[1]+p[2]}\right)==17.3611\,p[2]\\
p[2]\left(15.625+\dfrac{15.625\left(1-(1-p[0]-p[1]-p[2])^4\right)\left(p[6]+\dfrac{p[6]}{1-\dfrac{15.625}{8.68056\,c+15.625\left(1-(1-p[0]-p[1]-p[2])^4\right)}}+p[7]+p[8]+p[9]+p[10]\right)}{p[0]+p[1]+p[2]}\right)==26.0417\,p[3]
\end{array}\right)$$

Както се вижда този формат е достатъчно удобен за да се проследи коректността на описание на уравненията, а също и тяхната алгебрична сложност.

Изчислението обаче се сблъска със следните проблеми. Системата уравнения и неизвестни трябва да е крайна, следователно трябва да се ограничим с дължината на

локалната опашка на задания за всеки възел. Само по себеси това ограничение е логично, но проблемът с Wolfram Matematika е, че успява да генерира решение за системи с около до 20-25 неизвестни – а това е твърде малка точност: съвременните клауд-сървери обработват десетки, стотици, а евентуално и хиляди задания едновременно (в зависимост от грануларността на услугите).

Освен това решението на система с около 20 неизвестни отнема на настолен компютър време от порядъка на 2÷4 минути. При едно изследване ще са ни необходими стотици такива решения, следователно времето за изчисления също не може да се каже, че е напълно удовлетворително. Впрочем тук проблемът е, че не открихме как да потиснем търсенето на решения в комплексното пространство. Така че системата генерира от порядъка на 70-80 набора комплексни решения за вектора неизвестни и винаги точно едно реално решение, за което привеждаме следния пример:

{p[0]→0.0832712,p[1]→0.184537,p[2]→0.204473,p[3]→0.151043,p[4]→0.135939,p[5]→0.122345,p[6]→0.0601631,p[7]→0.0295851,p[8]→0.0145484,p[9]→0.00715407,p[10]→0.00351797,p[11]→0.00173004,p[12]→0.000850891,p[13]→0.000418591,p[14]→0.000206039,p[15]→0.000101478,p[16]→0.0000499959,p[17]→0.0000245057,p[18]→0.0000125238}

При увеличаване на броя неизвестни решението става числово нестабилно, тъй като продължава с часове и дни, и в крайна сметка приключва с грешка по препълване на паметта.

Все пак не е изключено изследователи с по-добър опит в дадена система за изчисления да имат по-добри резултати.

За решаване на така формулираните модели използваме предложевия в [Hasselt] числов метод.

5.1. Числов метод за итеративно решаване на модела

Тук ще опишем числов метод, който с малък брой итерации достига до приблизително решение на системата (e1) ÷ (e10), при това с много голяма точност и със съвсем малка вероятност за числова нестабилност.

Планът за нашето числово решение е следния. Ограничаваме системата уравнения до (e1) ÷ (e5), които са линейни уравнения спрямо неизвестните $p_0 \div p_{T_2}$. Относно останалите неизвестни $p_{T_2+1} \div p_\infty$ ще отбележим, че те са елементи на геометрична прогресия с първи елемент p_{T_2} и следователно могат лесно да бъдат изчислени. Освен това всъщност по-нататък във формулите ни е необходима само тяхната сума P_S. Освен $p_0 \div p_{T_2}$ системата (e1) ÷ (e5) има още две неизвестни и това са трансферните темпове σ и ρ. Можем да оценим приблизителните стойности на тези темпове с цел да ги използваме при първата итерация. Полученото решение на системата използваме за корекция на тези параметри, които използваме за контролни. Изпълняваме следващите итерации аналогично, докато достигнем достатъчно точни равновесни стойности на контролните параметри.

Така можем да формулираме следната ***итеративна процедура***.

Стъпка 0. Входни стойности: n, c, Λ, $T_{1,2}$, Φ;

Деривати: $\lambda := \Lambda/n$, $\omega := \Lambda/nc\Phi$, k:=1; /∗ k е номер на итерация

Апроксимации: $\rho^{[0]} := c\omega\Phi - \lambda$, (i1)

$\sigma^{[0]} := \lambda/\Phi - c\omega$; (i2)

Преход към Стъпка k.

Стъпка k. Решаваме (e1) ÷ (e5) спрямо $p_0 \div p_{T_2}$, използвайки $\sigma^{[k-1]}$ и $\rho^{[k-1]}$; (i3)

Изчисляваме $\sigma^{[k]}$ и $\rho^{[k]}$ използвайки зависимостите (e6) и (e9);

Ако ($|\sigma^{[k]} - \sigma^{[k-1]}| < 1\%$ and $|\rho^{[k]} - \rho^{[k-1]}| < 1\%$): преход към Край; (i4)

k++;

Преход към Стъпка k.

Край.

Основание за (i1) при определяне на началното приближение $\rho^{[0]}$ е наблюдението, че съгласно условието за ергодичност равновесната стойност на темпа на постъпване на

N	*c*	*v*	T1	T2	*n*	Λ	Φ0 = Λ/ncω	λ	ω	ρ0	σ0	(λ+ρ0)/ω					
	2	**4**	3	5	**200**	**8000**	**0.9**		40	22.2	20	*36*	2.7					
	Σπ0	π0	π1	πc	πT1-1	πT1	πT2	πT2+1	PR	PN	PS	PΣR	ρ	σ	£	*T*	ε	Θ%
0	7.3	0.136	0.368	0.5	0.496	0.526	0.426	0.503	0.356	0.45	0.19	0.83	3.395	*6.325974*	4.71	0.02	34.57	9.6
1	4.9	0.206	0.402	0.39	0.392	0.526	0.426	0.212	0.347	0.25	0.4	0.82	15.09	*13.09763*	5.99	0.04	22.51	46
2	6.6	0.153	0.378	0.47	0.469	0.526	0.426	0.305	0.315	0.35	0.34	0.78	11.31	*10.5317*	5.45	0.03	28.21	28
3	6	0.167	0.386	0.45	0.446	0.526	0.426	0.272	0.323	0.32	0.36	0.79	12.69	*11.38947*	5.6	0.03	26.39	33
4	6.2	0.162	0.384	0.45	0.455	0.526	0.426	0.284	0.32	0.33	0.35	0.79	12.23	*11.08802*	5.55	0.03	27.05	31
5	6.1	0.164	0.385	0.45	0.452	0.526	0.426	0.28	0.321	0.32	0.36	0.79	12.4	*11.19212*	5.57	0.03	26.82	31
6	6.1	0.163	0.384	0.45	0.453	0.526	0.426	0.281	0.32	0.33	0.35	0.79	12.34	*11.15589*	5.56	0.03	26.9	31
7	6.1	0.163	0.384	0.45	0.453	0.526	0.426	0.281	0.321	0.32	0.35	0.79	12.36	*11.16845*	5.56	0.03	26.88	31
8	6.1	0.163	0.384	0.45	0.453	0.526	0.426	0.281	0.32	0.32	0.35	0.79	12.36	*11.16408*	5.56	0.03	26.89	31
9	6.1	0.163	0.384	0.45	0.453	0.526	0.426	0.281	0.321	0.32	0.35	0.79	12.36	*11.1656*	5.56	0.03	26.88	31
10	6.1	0.163	0.384	0.45	0.453	0.526	0.426	0.281	0.321	0.32	0.35	0.79	12.36	*11.16507*	5.56	0.03	26.88	31

Фигура 5.1. Електронна таблица на итеративната числова процедура с 11 стъпки.

задания от съседи-донори заедно с темпа λ на централизирано разпределени задания би трябвало да изравнява темпа на обслужване, който максимално е $c\omega$, умножен с коефициента на глобалното натоварване Φ[1]. Подобно е и основанието за (i2) при определяне на началното приближение $\sigma^{[0]}$.

Формулировката на (i3) предписва решаване на система линейни уравнения на всяка стъпка от итерацията, което в общия случай макар и тривиална, би могло да бъде бавна за изпълнение задача. На практика обаче е напълно възможно зависимостите между $p_0 \div p_{T_2}$ да бъдат изведени предварително и на всяка итерация да бъдат изчислявани директно като алгебрични изрази. В този случай също извеждаме само „особените" вероятности p_0, p_c, p_{T_1} и p_{T_2}.

От опита ни с тази итеративна числова процедура можем да направим следните изводи. Процедурата е бързо сходяща и на практика не е необходимо да се въвежда условие за край (i4), тъй като гарантирано в рамките на 5-6 итеративни стъпки наблюдаваните параметри влизат в необходимата област на равновесните си стойности. Поради това изпълнението на процедурата може да се оформи като електронна таблица на Excel, всеки ред от която съдържа последователна итерация на процедурата, а

[1] (i1) всъщност би било точно уравнение, а не приближение, ако вместо Φ можехме да използваме отношението между броя донори и броя приемници в системата.

колоните – съответните параметри – междинни и крайни. На фиг. 5.1. е представено използването на такава таблица. Входните параметри (в бежовите клетки), техните деривати и началните апроксимации са изведени над таблицата. След това под реда с етикети са разположени редовете с фиксиран брой итеративни стъпки (11). Ако проследим колоната на който и да параметър – например на σ (маркирано в сиво) – ще видим, че само за 5-6 итерации този параматър се доближава до равновесна стойност.

Друго наблюдение е, че числовата процедура е сравнително нечувствителна към стойностите на началните апроксимации и освен това проявява значителна числова устойчивост. Много рядко – примерно под 5% – числовият експеримент завършва с грешка и в такъв случай е достатъчно да се промени минимално някой от входящите параметри, за да получим търсената стойност. Например ако за входния набор параметри на фиг. 5.1. получим грешка, би било достатъчно да сменим да кажем Λ от 8000 задания в секунда на примерно 7999 и ще получим необходимите характеристики на дадената система при практически същите условия.

За устойчивостта и общата коректност на процедурата и направения анализ може да се съди също по факта, че повечето от междинните и крайни стойности са вероятности, които имат тесен числов обхват (0, 1). Наличието на грешка в който и да е етап на анализа води до стойности извън този интервал. Също така моделът се подчинява на условия за обща логичност: така например при натоварване на системата Φ=90% е логично да очакваме вероятността за състояния донор или неутрален на произволен възел да бъде по-голяма от тази за състояние приемник и др.т.

5.2. Решаване на по-сложни модели с използване на агрегирани състояния

Ако се върнем към диаграмите на фиг. 3.1. ще видим, че за някои случаи на СВ тези диаграми могат да представят сравнително по-висока сложност за директно алгебрично решение на съответната система уравнения от стъпка (i3)[1]. Например могат да бъдат

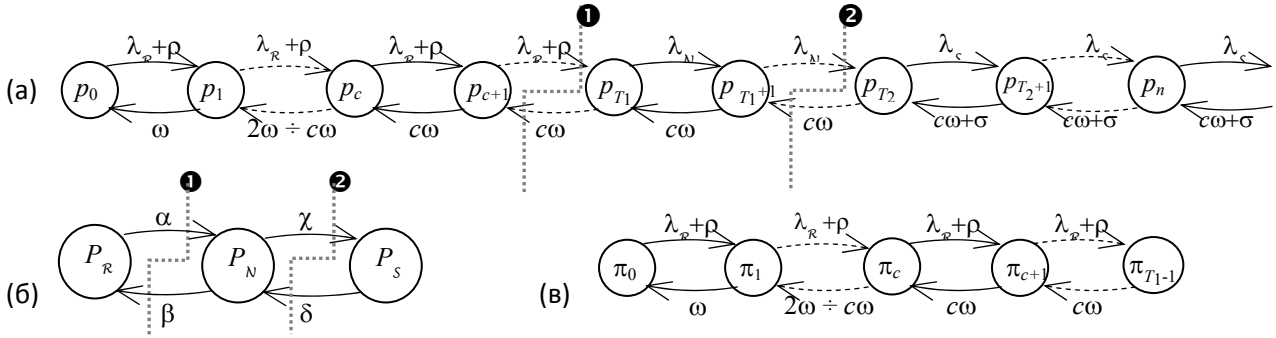
[1] сложните алгебрични изрази е желателно да се избягват поради възможността за техническа грешка при извода им, а също и при въвеждането им в електронната таблица.

добавени повече състояния на обслужващия възел при еднаква дължина на опашката за задания. Освен това реалистично е темповете на обработка да бъдат функция на състоянието на възела, тъй като с нарастване на броя задания за локално обслужване и[ли] за балансиращ обмен с други възли в клъстера се увеличава натоварването на локалния планиращ процес, който естествено има приоритет пред обслужващите процеси.

Също така асинхронните (Пуасонови) заявки за достъп до мултимедийни данни формират [дълга] последователност от периодични под-заявки. Възможни са и заявки за синхронно изпълнение на група услуги или паралелно изпълнение на реплики на една услуга. В тези случаи моделираната система е мрежа от n $\mathrm{G/G}/c$ обслужващи възли. Анализът на такива случаи използва различни техники, най-представителна колекция от които е направил Клайнрок в Queuing Systems I [14].

Тук представяме един вариант на декомпозичионен анализ, при който състоянията на възлите се агрегират в няколо групи. В резултат анализираме опростена система с агрегирани състояния, след което извършваме анализ на базовите състояния, от които е съставено всяко от агрегираните състояния.

Ще представим анализа на системата от фиг. 3.1.(а), за която обаче центаралния балансиращ процес разпределя потока от задания за обслужване в зависимост от локалното състояние на възлите – т.е. със съответните $\lambda_{\mathcal{R}|\mathcal{N}|\mathcal{S}}$ – ***фиг. 5.2.(а)***.



Фигура 5.2. (а) Диаграма на възел с частично централизирано балансиране ($\lambda_{\mathcal{R}|\mathcal{N}|\mathcal{S}}$); (б) диаграма на възела с агрегирани състояния; (в) диаграма на агрегираното състояние приемник $P_{\mathcal{R}}$.

За модела на такъв възел е в сила вече изведената система уравнения (е1) ÷ (е10), а също и останалите зависимости до (е23), но с необходимите корекции на λ-коефициентите. Например аналогът на (е18) има следната форма:

$$\delta = (\lambda_{\mathcal{R}}+\rho)p_{T_1\text{-}1} + \lambda_{\mathcal{S}}p_{T_2} + c\omega p_{T_1} + c\omega p_{T_2}. \qquad \text{(a1)}$$

След като разполагаме с такова аналитично представяне на модела, можем да пристъпим към неговия анализ чрез агрегирани състояния. Той се извършва на две стъпки. При първата стъпка агрегираме по подходящ начин моделните състояния да система с ограничен брой агрегирани състояния, която анализираме за ад определим вероятността (т.е. дела време), с която възелът се намира в съответното агрегирано състояние. Следващата стъпка е да се анализира разпределението на вероятностите на отделните състояния, формиращи съответното агрегирано състояние. Крайното решение на системата е произведение на тези вътрешни вероятности със съответната агрегирана вероятност.

И така ако разгледаме диаграмата на фиг. 5.2.(а), една очевидна възможност за декомпозиция на системата е на четири части, всяка от които съответства на една от групите уравнения (е1) ÷ (е4). Така анализът на всяка група се свежда до „табличен” случай на Марковска верига. Например (е1) е $\mathrm{M/M}/c/c$-модел, (е2) е $\mathrm{M/M/1}/(T_1\text{-}c)$-модел, (е3) е $\mathrm{M/M/1}/(T_2\text{-}T_1)$-модел и (е4) е $\mathrm{M/M/1}$. Всъщност за да запазим връзката със семантиката на моделирания обект, без при това да усложним анализа, извършваме декомпозицията на само три агрегирани състояния, всяко от които съответства на състояние на възела: приемник, неутрален и донор – ***фиг. 5.2.(б)***. Тук ще приведем само анализа на най-сложното агрегирано състояние – $P_{\mathcal{R}}$, което обединява (е1) и (е2).

Разглеждаме $P_{\mathcal{R}}$ като изолирана система, съставена от T_1+1 на брой състояния с вероятности π_i – ***фиг. 5.2.(в)***. В този случай, като имам предвид „табличните” решения за $\mathrm{M/M}/c/c$- и $\mathrm{M/M/1}/(T_1\text{-}c)$-модели:

$$\pi_i = \begin{cases} \frac{1}{i!}\left(\frac{\lambda_R + \rho}{\omega}\right)^i \pi_0, i \le c \\ \frac{c^c}{c!}\left(\frac{\lambda_R + \rho}{\omega}\right)^i \pi_0, c < i \le T_1 - 1 \end{cases} \tag{a2}$$

където

$$\pi_0 = \frac{1}{\sum_{i=0}^{c} \frac{1}{i!}\left(\frac{\lambda_R + \rho}{\omega}\right)^i + \frac{c^c}{c!}\left(\frac{\lambda_R + \rho}{\omega}\right)^c \left[\frac{\left(\frac{\lambda_R + \rho}{c\omega}\right)^{T_1 - c} - 1}{\frac{\lambda_R + \rho}{c\omega} - 1} - 1\right]} \tag{a3}$$

В (a4) сумата $\sum_{i=0}^{c} \frac{1}{i!}\left(\frac{\lambda_R + \rho}{\omega}\right)^i$ може да бъде решена до алгебричен израз чрез използване на познатия метод на z-преобразованието, който е най-добре обяснен в [5]. Всъщност за неголеми стойности на c (а да не забравяме че това е броят паралелно работещи процесори или ядра във всеки възел – типично 1, 2 или 4) можем лесно да кодираме този израз и без алгебричната форма чрез `if (c=1|2|4)`…-израз в електронната таблица[1].

По същия начин извеждаме и изразите за π_i на агрегираните състояния P_N и P_S.

След като изчислим всички локални π_i, пристъпваме към агрегационнта фаза, където трябва да определим дела на агрегираните състояния P_R, P_N и P_S в цялата система. Работим по диаграмата на фиг. 5.2.(б):

$$P_R = \frac{1}{1 + \frac{\alpha}{\beta} + \frac{\alpha\chi}{\beta\delta}}, \tag{a4}$$

$$P_N = \frac{\frac{\alpha}{\beta}}{1 + \frac{\alpha}{\beta} + \frac{\alpha\chi}{\beta\delta}}, \tag{a5}$$

$$P_S = \frac{\frac{\alpha\chi}{\beta\delta}}{1 + \frac{\alpha}{\beta} + \frac{\alpha\chi}{\beta\delta}}, \tag{a6}$$

[1] за тези малки стойности на горния индекс на сумата `if`-изразът ще е с по-малка сложност, отколкото евентуалната му алгебрична форма след право и обратно z-преобразование.

където

$\alpha = \pi_{T_1\text{-}1}(\lambda_{\mathcal{R}} + \sigma)$, (от „разрез” ❶ на фиг. 5.2.) (a7)

$\beta = \pi_{T_1} c\omega$, (от „разрез” ❶ на фиг. 5.2.) (a8)

$\chi = \pi_{T_2\text{-}1}\lambda_{\mathcal{N}}$, (от „разрез” ❷ на фиг. 5.2.) (a9)

$\delta = \pi_{T_2} c\omega$. (от „разрез” ❷ на фиг. 5.2.) (a10)

Накрая всяка необходима „особена” вероятност може да изчислим по следните формули:

$p_i = \pi_i P_{\mathcal{R}}$ за $0 \le i \le T_1\text{-}1$, (a11)

$p_i = \pi_i P_{\mathcal{N}}$ за $T_1 \le i \le T_2\text{-}1$ и (a12)

$p_i = \pi_i P_{\mathcal{S}}$ за $T_2 \le i < \infty$. (a13)

След тези изводи стигаме до следния план за прилагане на итеративната процедура от предходната точка 5.1.:

1) изчисляваме $\pi_{T_1\text{-}1}$ от (a2) и (a3); аналогично от съответните формули за $P_{\mathcal{N}}$ и $P_{\mathcal{S}}$ (които не сме представили тук) изчисляваваме π_{T_1}, $\pi_{T_2\text{-}1}$ и π_{T_2} (за началната итерация са ни необходими $\sigma^{[0]}$ и $\rho^{[0]}$ примерно от (i1) и (i2) с точност до $\lambda_{\mathcal{R}|\mathcal{N}|\mathcal{S}}$;

2) изчисляваме α, β, χ и δ от (a7) ÷ (a10);

3) изчисляваме $P_{\mathcal{R}}$, $P_{\mathcal{N}}$, и $P_{\mathcal{S}}$ от (a4) ÷ (a6);

4) изчисляваме $\sigma^{[\text{next iteration}]}$ от (e9) и $\rho^{[\text{next iteration}]}$ от (e6);

5) изчисляваме необходимите моделните резултати – например (e15), (e19), (e20) и др. т.;

6) повтаряме 1) ÷ 5) до постигане на желаната точност.

Стъпки 1) ÷ 5) се оформят като един ред от електронна таблица. Както вече отбелязахме, за достигане до равновесни стойности са достатъчни десетина реда, всеки от които е копие от предишния и наследява неговите итеративни резултати.

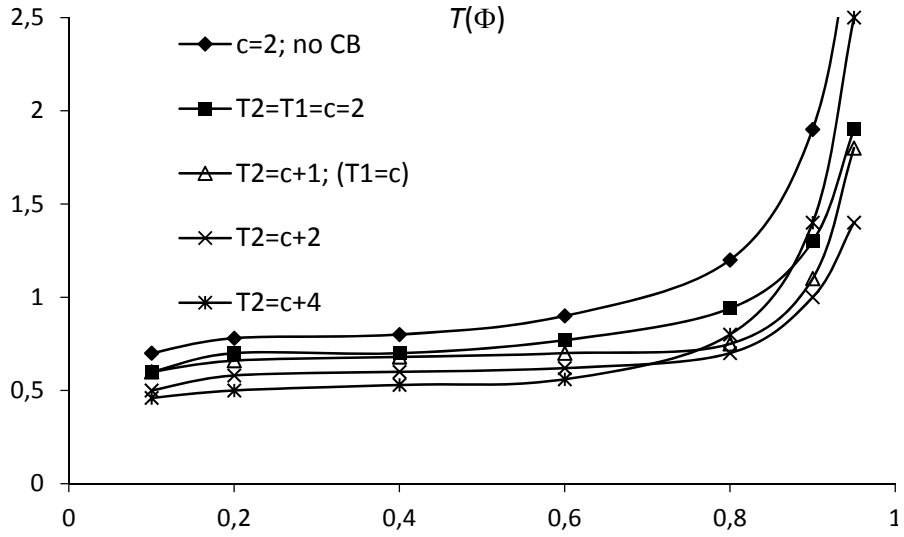
Описаният тук метод с агрегирани състояния е математически еквивалентен на общото решение от т. 5.1., но помага за достигане на по-компактни формули, поради което е по-защитéн от грешки, включително и поради това, че грешките по-лесно се локализират.

След като се постараем да изведем необходимите формули, както и да ги кодираме в електронната таблица – дейности, които обичайно отнемат около една-две седмици – можем да получаваме моделни резултати за всяка комбинация на входните параметри от огромното моделно пространство но СВ практически моментално. А това не е възможно по който и да е от другите налични методи – симулационно моделиране, експериментален тест или дори използване на изчислителна система от типа на Wolfram Matematika, при които за всеки входен набор параметри се изчаква определен период от време до получаване на резултат – както отбелязахме от поне няколко минути като минимум.

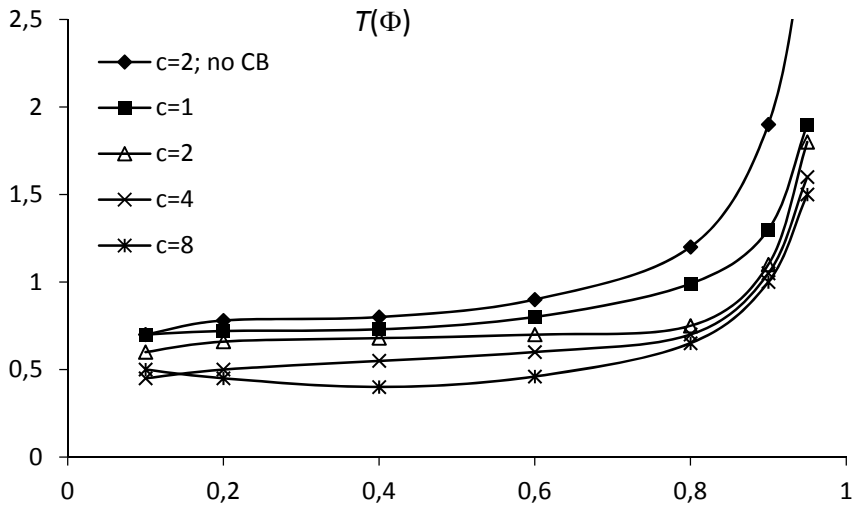
6. Зависимости между моделираните системни параметри

Ще представим няколко характерни случая на обслужване с клауд-инфраструктура, изследвани чрез представените тук модели.

Нека разгледаме най-напред базовата характеристика – зависимостта на времето на обслужване на заданията T от глобалното натоварване на системата Φ. На ***фигура 6.1.*** са представени резултатите за тази зависимост за случай $\mathbb{BR}$ при различни прагове на точност $T_{1,2}$ и паралелизъм на обслужващите възли c=2. Най-добрите резултати на тази характеристика в са постигнати при параметри T_1= c =2 и T_2 > T_1. Бързодействието на системата при ниско и умерено натоварване (<70%) е най-добро за T_2=6, а при по високо натоварване – за T_2=4. Като референтна стойност е представена характеристиката на



Фигура 6.1. Зависимост на времето за обслужване T от натоварването на клауд-клъстера Φ при различни прагове на точност $T_{1,2}$ и паралелизъм на обслужващите възли c=2.

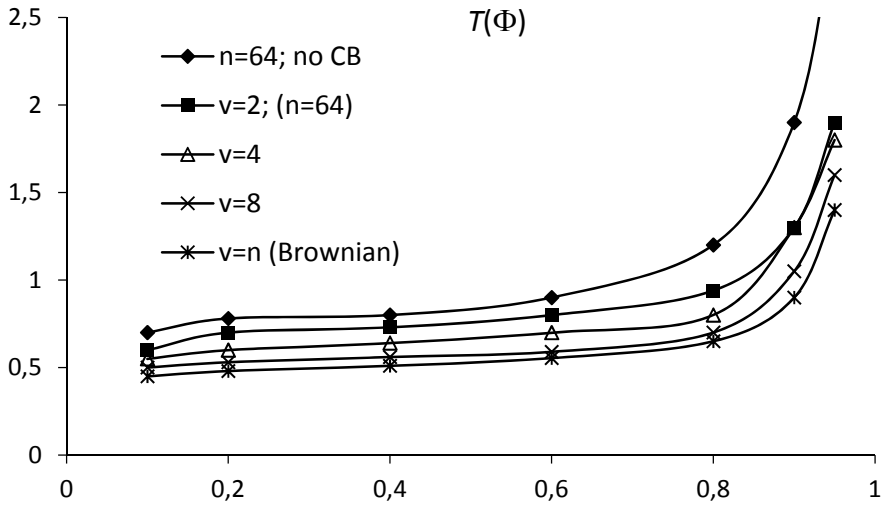


Фигура 6.2. Зависимост на времето за обслужване T от натоварването на клауд-клъстера Φ при различен паралелизъм на обслужващите възли c.

системата при изключени балансиращи процедури (т.е. $\sigma = \rho = 0$). Високи стойности на T_2 (=6) водят също така до влошаване на производителността при високо натоварване >80, включително и спрямо балансираща схема без междинно състояние – т.е. при T_1=T_2.

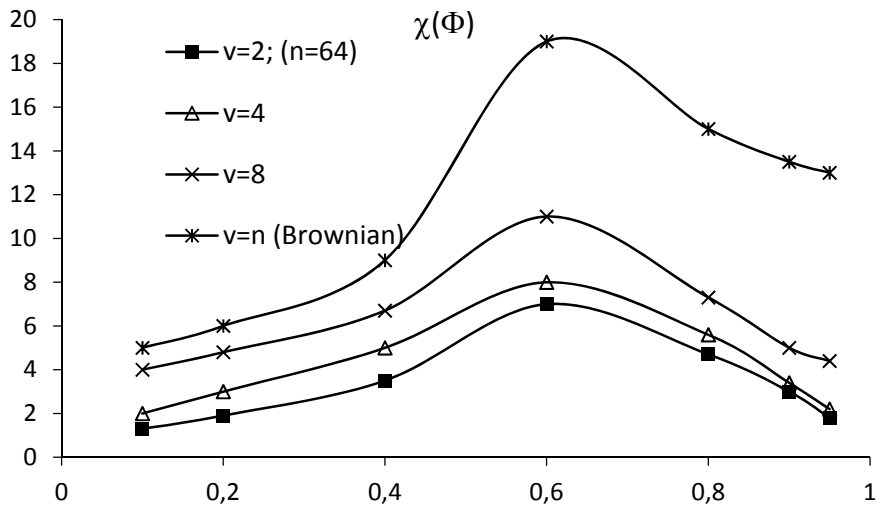
Тези резултати автоматично повдигат въпроса за влиянието на паралелизма на обслужващите възли върху бързодействието на системата. На фиг. 6.2. са представени функциите $T(\Phi)$ за случай $\mathbb{BR}$ при вариращ брой процесорни ядра c. Получените характеристики показват, че високият машинен паралелизъм е благоприятен за бързодействието, но дори еднопроцесорен възел с балансиране има по-добро бързодействие от двупроцесорен възел без балансиране.

Нека сега разгледаме и дифузионни случаи. Докато при $\mathbb{BR}$-случай локалността е v=n, при $\mathbb{DF}$-случай v<<n. На ***фигура 6.3.*** е представена зависимостта на времето за обслужване T от натоварването на клауд-клъстера Φ при дифузионни балансиране с различен брой съседи на възлите v. При този модел на балансиране бързодействието на системата е по-добро при по-висок брой на възлите-съседи и разбира се най-добра е



Фигура 6.3. Зависимост на времето за обслужване T от натоварването на клауд-клъстера Φ при дифузионни балансиране с различен брой съседи на възлите – локалността v.

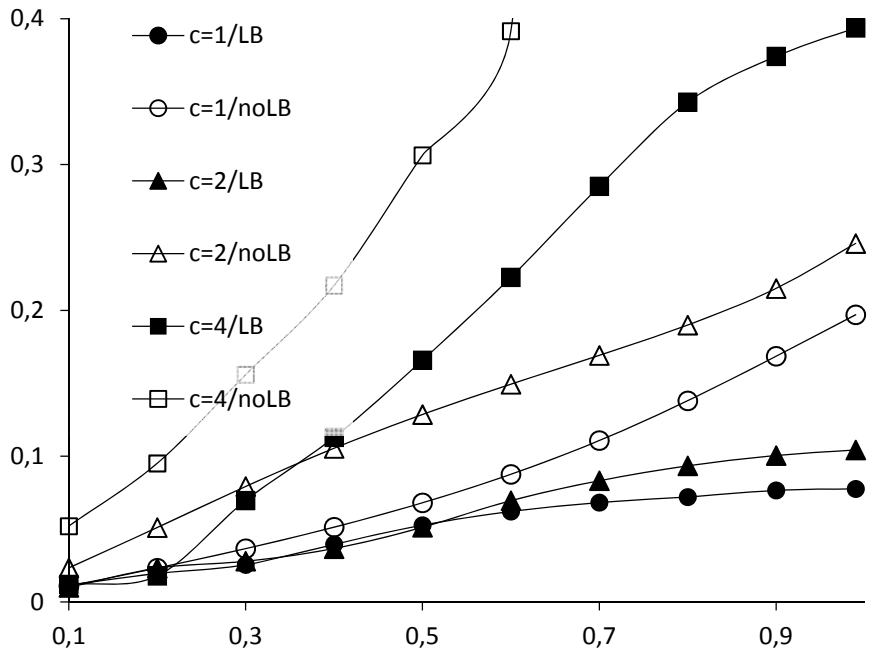
характеристиката за максималната стойност на този параметър – v=n. Трябва да се има предвид обаче, че изследваните модели не отчитат въздействието на генерирания системен свръхтовар от балансиращите процедури. За да отчетем това въздействие е необходимо в моделите на фиг. 3.1., на базата на които са получени тези резултати, да се направят изменения, подобни на тези от фиг. 5.2., така че темпът на обслужване на задания вместо константа да стане функция на други параметри включително и системния свръхтовар. В такъв случай може да се очаква, че основното предимство на дифузионните схеми пред Брауновите – очакваното намаляване на броя обменяни съобщения между възлите – ще се отрази на бързодействието на системата за обслужване. От друга страна ако съпоставим характеристиките за v=8 и v=n (=64), виждаме колко близки са тези стойности на времето за обслужване T и следователно можем да очакваме, че в някои случаи след като отчетем фактора свръхтовар дифузионните схеми имат предимство пред Брауновите.



Фигура 6.4. Зависимост на глобалния темп на обмен на съобщения от натоварването на системата $\chi(\Phi)$ при различен брой съседи на възлите – локалността v.

Във връзка с току-що отбелязаното нека изследваме именно системния свръхтовар, измерен чрез глобения темп на обмен на съобщения при различно натоварване на системата $\chi(\Phi)$. Тази характеристика е представена на ***фиг. 6.4.*** Основният извод, който може да се направи от получените характеристики е, че действително дифузионните балансиращи схеми са много по-„икономични" в сравнение с Брауновите от гледна точка на генериран системен свръхтовар. Това съотношение е най-силно изразено при средни и високи стойности на натоварването – 40-90%, като по принцип натоварване около 60% е условие за максимален свръхтовар при всички схеми – очевидно при това натоварване възлите най-често сменят състоянието си.

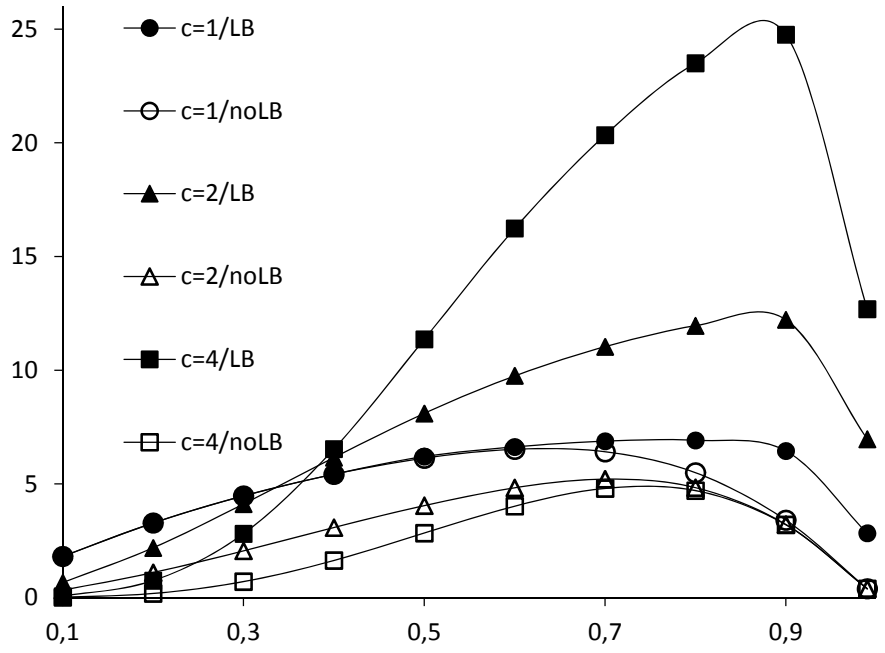
Нека сега разгледаме характеристики на клауд-обслужването, получени чрез анализ на по-„сложните" модели от фиг. 5.2. Представените тук резултати се отнасят за клъстер от 800 обслужващи процесора, но организирани в архитектура с различен модел на конкурентостта:



Фигура 6.5. Средно време на обслужване при различно системно натоварване $T(\Phi)$.

- разпределена система от 800 еднопроцесорни възела: c=1;
- системата се състои от 400 двупроцесорни възела (n=400, c=2);
- система от 200 четирипроцесорни възела (n=200, c=4).

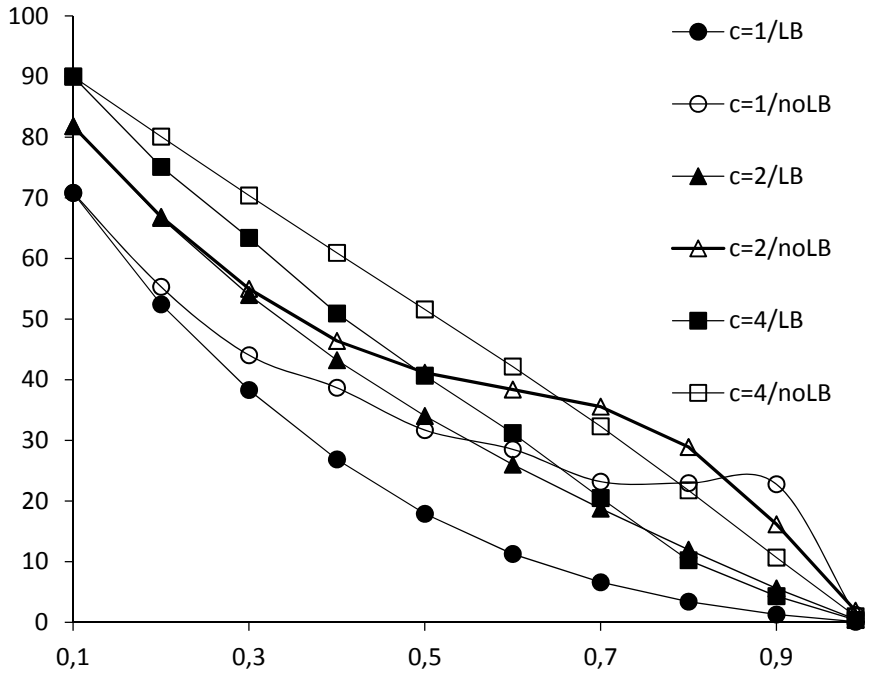
Всяка от тези системи обслужва Пуасонов процес на постъпване на асинхронни задания с темп $\Lambda =$ 8000 в минута. Локалността на възлите при тези числови експерименти е фиксирана на v = 4, което съответства на логическата топология примерно на четиридименсионен хиперкуб или двудименсионна решетка – тор. Ще проследим зависимостите на три параметъра на обслужването от системното натоварване: средното време на обслужване T, темп на получени системни съобщения ε от съседните възли и вероятността Θ за неактивно едно или повече ядра в обслужващите възли.



Фигура 6.6. Зависимост на темпа на получаване на системни съобщения от съседни възли от системното натоварване $\varepsilon(\Phi)$.

Средното време на обслужване при различно системно натоварване $T(\Phi)$ е представено на диаграмата на ***фиг. 6.5***. Всеки от трите случая на локален паралелизъм е съпоставен с вариант на клауд без товарно балансиране. Очаквано „балансираните" системи обслужват много по-бързо от небалансираните – особено при високо натоварване от 90-95%. Показателно е, че балансирането променя характера на зависимостите от суперлинейни към сублинейни. Характерно е, че най-нископаралелният случай – система с 800 еднопроцесорни сървера – дава най-добра производителност, докато четирипроцесорните сървери са (относително!) най-бавни. Обяснението на този резултат е, че разгледаните от нас модели не отчитат локалното балансиране, което планиращия процес на операционната система извършва във всеки обслужващ възел.

По отношение на зависимостта на темпа на получаване на системни съобщения от съседни възли $\varepsilon(\Phi)$, представена на диаграмата на ***фиг. 6.6***. виждаме, че най-благоприятен (т.е. нисък) е този параметър при системата с най-малък брой процесори



Фигура 6.7. Зависимост на вероятността за неактивно едно или повече ядра в обслужващите възли от системното натоварване $\Theta(\Phi)$.

във възел. В този случай сме оставили характеристиките на клъстер без балансиране, макар че в него не биха се обменяли такива съобщения; тези характеристики показват, че небалансираните възли по-рядко променят състоянието си, отколкото балансираните. Нека проследим какво отношение има тази „стабилност" към общата ефективност на обработката.

Ефективността на обслужването в моделираната инфраструктура можем да оценим чрез вероятността Θ за неактивно едно или повече ядра в обслужващите възли – ***фигура 6.7***. Представената вероятност е обикновена сума на вероятностите на всички състояния p_i, $i \in [0, c)$, но за по-голяма прецизност би могло различните състояния да се сумират със съответен коефициент на тежест. Най-ефективна е обработката с еднопроцесорни/едно-ядрени сървери. Този извод едва ли може да се генерализира без допълнително изследване и модификации на представените тук модели. Но той е логичен в

съответствие с казаното по отношение не резултатите за времето на обслужване T. Небалансираните възли по-често изпадат в състояние на неработещи ядра. Следователно можем да заключим, че тяхната стабилност в смисъл на по-редки преходи между праговите състояния, не допринася за ефективността на обработка, а ги подържа във състояния, при които в дадени възли има необработващи процесорни ядра, докато в други заданията чакат обслужване на дълги системни опашки.

7. Обобщение

Представените в това изследване резултати се абстрахират от конкретни обслужващи среди, а по-скоро илюстрират подхода за анализ и възможностите за оценка на параметрите на обработка в разпределената сървърна инфраструктура. Ако се опитаме да съпоставим параметризацията на тези модели – както по отношение на обслужващия процес, така и по отношение на модела на натоварване – лесно ще установим, че тя е по-изчерпателна и детайлна в сравнение с прегледаните в табл. 1.1./2. модели. При това засега не сме проверили какво въздействие върху производителността биха имали различни интригуващи детайли на моделите – да припомним например дори само въпроса с хистерезисната или транзитната инверсия. В известен смисъл сме се самоограничавали да изчислим повече възможни варианти на тези модели, тъй като същественото на този етап на анализа е не толкова изследване на повече области от обширното моделно пространство, а по-скоро верификация на моделните резултати чрез експерименти в реална обслужваща инфраструктура.

Извън принципната задача за верификация, нека се спрем на възможните развития на тези модели. На първо място това е разнообразяване на изследваното натоварване. Представените тук резултати се отнасят към класическия случай на употреба на ИТ-инфраструктурата за асинхронни информационни задания. Когато обаче заданието е за достъп до поточни данни – най-характерният случай за които са ***мултимедийните данни*** – тогава еднократната клиентска заявка планира [дълга] последователност от квази-периодични заявки за изосинхронен обмен. Очевидно за моделиране на мултимедийни приложения такъв процес на задания трябва да се добави като съществен компонент към представения тук асинхронен процес. Друг модел на обслужване биха представлявали

задания за ***паралелна*** или ***потокова обработка***. Те не са типична масова услуга за клауд-инфраструктурата и причината за това (освен по-високата степен на специализация) е, че такива задания изменят съществено обичайните параметри на работа на планиращия процес, с което разбира се не подобряват цялостното качество на обслужване и ефективността. Именно поради това те струват и по-скъпо на съответните потребители (в сравнение с масовите случаи на употреба). Те могат да се моделират като групови или групово-периодични заявки за обслужване. Впрочем по отношение на конкурентната обработка с паралелизъм по данни сме представили резултати в [8, 9], които също са моделирани чрез системи за масово обслужване, но техният анализ се базира на компютърни симулации.

Това, което може непосредствено да се изследва с така изведените модели от 4. и 5. глава са още няколко важни параметъра на обслужването. Например ***грануларността*** на обработваните задания може да се моделира чрез едновременно увеличение на стойностите на Λ и ω при запазено натоварване Φ, което би съответствало на по-фина грануларност на заявките за обслужване, а промяната им в обратна посока – съответно на по-едра.

Ускорението $S_{n,c}$ и ***ефективността*** $E_{n,c}$ – които не сме коментирали тук – също могат лесно да се изведат и анализират на базата на средното време за обслужване, при това в контекста на локалния паралелизъм c на възлите:

$$S_{n,c} = T(n,c)/T(1,1), \tag{o1}$$

$$E_{n,c} = S(n,c)/(n*c). \tag{o2}$$

Влиянието на праговете на точност на наблюдението $T_{1,2}$ (на фиг. 6.1.) и локалността на възлите v (т.е. логическата топология, на фиг. 6.3.) върху производителността на системата ни дава пряко основание за въвеждане на ***адаптивност*** на тези параметри на балансиращите схеми в съответствие с текущото натоварване и други системни параметри.

Освен изброените непосредствени възможности за допълване на изследването, има също така възможности за развитие на представените модели. Те са основно в две посоки – векторизация на някои скаларни параметри и корелация между независими параметри на модела. ***Векторизацията*** би позволила анализ на комплексни параметри на качеството на обслужване, например съгласното отбелязаното в гл. 1. цената на обслужване, чиято класическа скаларна мярка е процесорното време[1], може да се разшири с паричен и „зелен" компонент – в системи, за които паричната и енергийната цена на процесорното време и комуникациите варира между възли и клъстери.

Отчитайки въздействието на системния свръхтовар лесно можем да допълним модела, като представим системните комуникации за самостоятелни събития на постъпване на задания за обслужване, задължително отчитайки обаче тяхната в общия случай значително по-фина грануларност спрямо тази на потребителските заявки. Друг подход за това е да въведем ***корелация*** между сега независими системни параметри: темпът на обслужване ω да стане функция на параметрите на балансиращия обмен ρ, σ и χ[2]. Освен очевидното локално ниво на тази корелация (т.е. в обработващия възел), тя може да има отражение и на глобално ниво тъй като клауд-клъстерите са мрежи с общодостъпно предаване и следователно комуникациите между всяка двойка или по-добре v-орка възли се конкурират с останалите за общия комуникационен ресурс.

Внимателният читател ще се сети също за споменаване в изложението на възможността моделът да се разшири в посока хетерогенност между възлите, но сега не бихме обсъждали тази тема за да не се създаде впечатление, че предстоящите изследователски задачи едва ли не започват да преобладават над вече извършеното.

---

[1] една практична дефиниция на цената на обслужване е например

$$\xi = T/\underline{\Theta}, \text{ където } \underline{\Theta} = \Sigma(1-i/c)*p_i, \forall i \in [0, c). \qquad (o3)$$

[2] в контекста на представения числов метод ρ и σ, макар и константи, варират между последователните числови итерации (вж. фиг. 5.1.), а $\chi=\chi(\Phi)$: фиг. 6.4.

Терминологичен речник

access point (WiFi)	станция за достъп
ad hoc	инцидентен, случаен
backbone network	опорна мрежа
background processing	[фонова |нископриоритетена] обработка
backup	архивиране
boot sector	системен [ОС-] сектор
bottleneck	тясно място
bridge (layer2 switch)	мост (форматен конвертор [на пакети])
broadband	широколентов (т.е. високоскоростен и високочестотен)
broadcast	общодостъпно предаване („един към всички”)
cache	вътрешна памет
collaborative computing	съвместна обработка
cookie	пакет с клиентски контекст
co-sheduling	синхронно планиране (на задания)
datawarehouse	склад за данни
desktop [|lightweight] grid	[настолен|лек] грид
distributed shared memory	разпределена обща памет (в разпределена система, мултикомпютър)
embedded system	[вградена I автономна] система
glass box	сива кутия
grid-aware application	грид-приложение
grid-unaware application	прозрачно грид-приложение ≡ разпределено приложение
header	етикет (на съобщение), заглавно поле

high-throughput computing, htc	пакетна фонова обработка
hop	междинен възел
host	главен компютър
hourglass principle (“All IP”)	принцип на пясъчния часовник
layer2 switch (bridge)	мост (форматен конвертор [на пакети])
leeching	егоистична употреба
lightweight grid	лек грид
load balancing	балансиране (на конкурентни ресурси)
locking	резервиране [на споделени ресурси]
log	дневник
mapping	картиране (планиране на заявки и ресурси)
marshaling	сериализиране
matchmaking	съчетаване (планиране на заявки и ресурси)
message passing	(синхронен) обмен на съобщения
message queuing	(асинхронен I буфериран) обмен на съобщения
middleware	междинен слой
mirror sites	репликирани Web услуги
mirroring	репликиране
multicast	групово предаване („един към няколко”)
nomadic user	случаен потребител (с ограничени права)
offline	отложен достъп [по заявка]
on-demand computing	обработка по заявка
online	пряк [интерактивен] достъп
over-the-air (OTA)	безжични телекомуникации (наземни или спътникови)
payload	[приложно] съдържание (на съобщение)
peer-to-peer (P2P)	всеки-към-всеки (равнопоставени комуникиращи процеси)
piggybacking	опаковано съобщение
polling	сканиране
precaching (prefetching)	предварително (буферирано) зареждане
proxy	заредим интерфейс

sampling	честота (темп) на сканиране
shared memory	обща памет (в паралелна система, мултипроцесор)
signaling	комуникационен свръхтовар (системен)
streaming	поточни данни
TOC (total cost of ownership)	себестойност на обслужването
token	купон, служебно съобщение
tolerance	допуск
trailer	закриващ етикет (на съобщение)
tuneling	вложено съобщение
unicast	единично предаване, 1:1
virtual hosting	сайтове с общ IP-адрес
virtual unicomputer	разпределен суперкомпютър
wizard	[интерфейс за] асистирана настройка
workflow	потокова композиция
workload [model]	модел на натоварването
disaster recovery	възстановяване след грешка

Литература

[1] Bahsoon, R. 2010. Towards a Framework for Dynamic Self-Optimization of Power and Dependability Requirements inCloud Architectures. To appear in Proceedings of the 4th European Conference on Software Architecture (ECSA 2010), Copenhagen, Denmark. LNCS, Springer.

[2] Bahsoon R. 2010. Green Cloud: Towards a Framework for Dynamic Self-Optimization of Power and Dependability Requirements in Cloud Architectures. In Proceedings of the ACM/IEEE 32nd International Conference on Software Engineering, 2-8. May 2010, Cape Town, South Africa.

[3] Cybenko G., 1994. Dynamic Load Balancing for Distributed Memory Multiprocessors. Journal of Parallel and Distributed Computing, 7, 1989, pp. 279 – 301.

[4] Eager, D., E. Lazovska, J. Zahorian. 1986. A Comparison of Receiver-Initiated and Sender-Initiated Adaptive Load Sharing. Performance Evaluation, Vol. 6., pp. 53-68, 1986.

[5] Geist, R., K. Trivedi. 1983. The Integration of User Perception in the Heterogenous M/M/2 Queue. Performace'83. pp. 203 – 216.

[7] ***Georgiev, V.*** Load Diffusion and Brownian Models for Cloud Balancing: between C-S and p2p. In Proceedings of ESM'2010, Hasselt, Belgium, 25th – 27th October, 2010. pp. 170 – 177. ISBN 978-90-77381-57-1.

[8] ***Georgiev, V.***, Getov, V. Simulation of replicated services in Jini. In: Tentner, Adrian, (ed.) Proceedings of the High Performance Computing Symposium - HPC 2002 : 2002 Advanced Simulation Technologies Conference, San Diego, California, Mission Valley Marriott, April 14-18, 2002. Simulation Councils, San Diego, USA, pp. 227-232. ISBN 1565552504

[9] ***Georgiev, V.***, Getov, V. Assignment Schemes for Replicated Services in Jini. In Proceedings of 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, Las Palmas de Gran Canaria, Spain, January 9-11, 2002, pp. 129-136, ISBN: 0-7695-1444-8.

[10] ***Georgiev, V.***, J. Karvo. Numerical Modeling of Load-balanced Multicore Servers in a Cloud Cluster. Proceedings of 12th Middle Eastern Simulation and Modeling Multiconference MESM'2011, Amman –Jordan, November 14-16., 2011. pp. 48 – 52. ISBN 978-90-7738167-0.

[11] Iosup Alexandru, Nezih Yigitbasi, and Dick Epema. On the performance variability of production cloud services. In CCGrid, pages 104 –113, may 2011.

[12] Karagiorgos, G., N. Missirli and F. Tzaferis. 2004. The generalized diffusion method for the load balancing problem. Advances in Parallel Computing, Volume 13, Elsevier B.V. 2004, Pages 225-232

[13] Kleinrock, L. History of the Internet and Its Flexible Future, IEEE Wireless Communications, February 2008, pp. 8-18.

[14] Kleinrock, L. Queueing Systems. Volume 1: Theory, New York: Wiley Interscience, 1975, pp. 417. ISBN 978-0471491101.

[15] Kuchen Herbert, Andreas Wagener. Comparison of Dynamic Load Balancing Strategies. In Proceedings of the SPDP - Symposium on Parallel and Distributed Processing, 1990.

[16] Kundu, Sajib; Rangaswami, Raju; Dutta, Kaushik; Zhao, Ming; Application Performance Modeling in a Virtualized Environment , High Performance Computer Architecture (HPCA), 2010 IEEE 16th International Symposium on , pp.1-14 Jan. 2010, http://dx.doi.org/10.1109 /HPCA.2010.5463058,

[17] Li, J., J. Chinneck, M. Woodside, M. Litoiu, G. Iszlai. Performance Model Driven QoS Guarantees and Optimization in Clouds. Proceedings of the ICSE Workshop on Software Engineering Challenges of Cloud Computing, IEEE Computer Society Washington, DC, USA 2009; pp. 15-22, ISBN: 978-1-4244-3713-9.

[18] Luis M. Vaquero, Luis Rodero-Merino and Rajkumar Buyya. 2011. Dynamically Scaling Applications in the Cloud. ACM SIGCOMM Computer Communication Review. 41/1, January 2011.

[19] MacVittie, L. 2010. Cloud Balancing: The Evolution of Global Server Load Balancing. F5 White Paper, 2010. (www.f5.com/pdf/white-papers/cloud-balancing-wp.pdf)

[20] Moschakis I., H. Karatza. Evaluation of gang scheduling performance and cost in a cloud computing system, Journal of Supercomputing, v. 59/2, February 2012; pp 975-992, Kluwer Academic Publishers Hingham, MA, USA. ISSN: 0920-8542 (print version), ISSN: 1573-0484 (electronic version).

[21] Oneil, D., H. Jeong , K. Jinoh , K. Kwon. Transport Layer Identification of P2P Super nodes, Proceedings of 26th IEEE International Conference on Computer Communications, April 2007, Barcelona, Spain, ISSN : 0743-166X, pp. 1-14.

[22] Xiong, K., H. Perros. Service Performance and Analysis in Cloud Computing. Proceedings of the 2009 Congress on Services (SERVICES '09), IEEE Computer Society Washington, DC, USA 2009; pp. 693-700, ISBN: 978-0-7695-3708-5.

[23] Xu, Jing; Â Zhao, Ming; Fortes, Jose; Carpenter, Robert; Yousif, Mazin; On the Use of Fuzzy Modeling in Virtualized Data Center Management, Proceedings of the Fourth International Conference on Autonomic Computing (ICAC), IEEE Computer Society, p. 25, June 2007, http://portal.acm.org/citation.cfm?id=1270385.1270747,

[24] Zhelev, R., ***Georgiev, V.*** A DHT-based Scalable and Fault-tolerant Cloud Information Service. Proceedings of the UBICOMM 2011: The Fifth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, Lisbon-Portugal, November 20-25, 2011., 2011. pp. 66 – 72. ISBN 978-1-61208-171-7.

[25] Zhelev, R., ***Georgiev, V.*** Resource Information Service for Cloud Datacenters. Proceedings of the International Conference on Information & Communication Systems, Irbid –Jordan, May 22-24., 2011. pp. 83 – 88. ISBN 978-1-4507-8208-1.

[26] Zhelev R. and ***V. Georgiev***. A Generic Resource Framework for Cloud Systems, Proceedings of The 4th International Conference on Distributed Computing and Grid-technologies in Science and Education, June 28 - July 3, 2010 Dubna, Russia., pp. 268 – 278. ISBN 978-5-9530-0269-1.

[27] Zhelev R. and ***V. Georgiev.***, Resource Abstractions in Cloud Frameworks, in the Proceedings of The 4th International Conference on Information Systems and Grid Technologies, Sofia, Bulgaria, 28th – 29th May, 2010. pp. 133 – 144.

Интернет източници

[28] Amazon. 2010. Amazon Elastic Compute Cloud (Amazon EC2) (http://aws.amazon.com/ec2/)

[29] Amazon. 2011. Amazon ElastiCache – Support in Four Additional Regions, CloudFormation Support, Free Webinar December 5, 2011 (http://www.webbox.org/2011/12/amazon-elasticache-support-in-four-additional-regions-cloudformation-support-free-webinar)

[30] Gong, Z.; P.Ramaswamy; X. Gu; and X. Ma. 2009. SigLM: Signature-Driven Load Management for Cloud Computing Infrastructures. 17th International Workshop on Quality of Service, IWQoS 13-15 July 2009. (http://ieeexplore. ieee.org/xpl/mostRecentIssue.jsp? punumber=5201378)

[31] SwiftWater. 2010. Cloud computing, load balancing, and extending the data center into a cloud. (http://vburke.wordpress. com/2010/03/26/cloud-computing-load-balancing-and-xtending- the-data-center-into-a-cloud/)

[32] Triebes K. 2010. Cloud Balancing: The Next Generation of Global Server Load Balancing. Virtualization Review (http://virtualizationreview.com/blogs/app-delivery-ondemand/2010/06/ cloud-balancing. aspx)

Списък на цитирания

7. цитирана в 1 публикация:

- Radko Zhelev. Blueprint of an Experimental Cloud Framework. Proceedings of the 6th International Conference on Information Systems and Grid Technologies, Sofia, Bulgaria, 1. – 3. June, 2012. pp. 338 – 345. ISSN 1314-4855.

8. цитирана в 1 публикация:

- Kirchev L. Simulation Modeling of Replicated Clustered Web Services. Proceedings of the First International Conference on Information Systems and Datagrid, Sofia, Bulgaria, February 17-18, 2005, pp. 141-146. ISBN: 954-649-761-4.

9. цитирана в 3 публикации:

- Ying Chen Lin; Sy-Yuan Li; Yuan-Shin Hwang; Dynamic load-balancing of Jini and .NET services. 2006 International Conference on Parallel Processing (ICPP'06), pp.-265. – 273.Columbus, Ohio. August 14 – 18. ISBN: 0-7695-2637-3
- Hewijin Christine Jiau, Chia Hung Kao. Dreamer: A resource management architecture for Jini federation, in Information and Software Technology, Volume 48, Issue 3, March 2006, Pages 170-186.
- Shih-Rung Lai, Extending Jini lease Concept on Service Resource Allocation Management, MSc Thesis, National Cheng Kung University Department of Electrical Engeneeering, Tanian Taiwan, R.O.C., June, 2003.

10. цитирана в 1 публикация:

- Radko Zhelev. Blueprint of an Experimental Cloud Framework. Proceedings of the 6th International Conference on Information Systems and Grid Technologies, Sofia, Bulgaria, 1. – 3. June, 2012. pp. 338 – 345. ISSN 1314-4855.

24. цитирана в 1 публикация:

- Georgi Pashov, Kalinka Kaloyanova. Requirements for Cloud Service Discovery Systems. Proceedings of the 6th International Conference on Information Systems and Grid Technologies, Sofia, Bulgaria, 1. – 3. June, 2012. pp. 280 – 293. ISSN 1314-4855.

25. цитирана в 1 публикация:

- Georgi Pashov, Kalinka Kaloyanova. Requirements for Cloud Service Discovery Systems. Proceedings of the 6th International Conference on Information Systems and Grid Technologies, Sofia, Bulgaria, 1. – 3. June, 2012. pp. 280 – 293. ISSN 1314-4855.

26. цитирана в 2 публикации:

- Radoslava Hristova. Monitoring of Business Processes in the EGI. Proceedings of the 6th International Conference on Information Systems and Grid Technologies, Sofia, Bulgaria, 1. – 3. June, 2012. pp. 294 – 301. ISSN 1314-4855.
- Goranova, R. D. Architecture of a SOA-based BPM Platform for EGI, To appear in Conference Proceedings of the 5th International Conference "Distributed Computing and Grid-technologies in Science and Education", 16 – 21. July, 2012, Dubna, Russia.