

**СИМУЛАЦИЯ НА ОТКРИВАНЕ НА РЕСУРСИ  
В ДИНАМИЧНА МРЕЖА****RESOURCE DISCOVERY SIMULATION IN GRID****Огнян Наков**

ФКСУ, Технически Университет - София

**Аделина Алексиева-Петрова**

ФКСУ, Технически Университет - София

**Пламенка Боровска**

ФКСУ, Технически Университет - София

**Милен Петров**

ФМИ, СУ „Св. Кл. Охридски”

**Abstract**

*The purpose of this paper is related to the increasing use of grid and communication difficulties that occurred in this area. The open issue is resource discovery finding the shortest path to reach a resource in dynamic grid where resources can be online or offline. In that paper we describe two methods which solving this problem: re-routing table and hybrid method. We design and implement application which simulates different number of grid nodes and applies those two approaches of resource discovery in order to study and improve this problem.*

**Keywords:** resource discovery, dynamical grid, re-routing tables, peer-to-peer

**ВЪВЕДЕНИЕ**

В множество сфери от дейността на човечеството е застъпено ефективното разпределение на ресурси за определени задачи, ограничени от конкретни времеви рамки.

Откриването на ресурси в динамична мрежа е един от актуалните и трудоемките процеси в този проблем. За избор на добър подход най-често се взимат в предвид изискванията за необходимото време и цена при използване на определени ресурси за изпълнение на дадена цел [1].

При един комплексен проект, съставен от множество зависими задачи, то оптималното време към стойностно разпределение на използваните ресурси е от огромно значение.

Проектирането и разработката на приложение за симулиране на динамични мрежи с различен на брой възли, поддържането на различни подходи за откриване на ресурси и графичното изображение на резултатите е от голяма важност за изследване и подобряване на този проблем.

**МЕТОДОЛОГИЯ**

В последните години голяма част от интереса се фокусира върху две системи: мрежи и peer-to-peer системите. И двете споделят една и съща основна идея – те са среди за споделяне на ресурси. Разликата между тях е, че са последвали различни еволюционни пътеки. Системите с мрежите се използват главно в сложни научни приложения, докато peer-to-peer системите са разработени около поточни услуги като споделяне на файлове [2].

Подходът за откриването на ресурси в динамична мрежа предполага, че ако се направи заявка в един от рутерите в мрежата за определен ресурс, ефективният механизъм за откриване на ресурси трябва да може да намери този специфичен ресурс. В най-простия случай рутерът проверява дали някой от локалните му ресурси отговаря на изискванията. Ако това не е изпълнено, той трябва да препрати заявката на своите съседи по случаен начин [3]. Това случайно препращане може да е задоволително в малка мрежа, но няма да е подходящо за нуждите на големи мрежи.

Един от методите, които се използва за да не се изпращат заявките в мрежата на случаен принцип, е механизма на маршрутизиращите таблици [4]. Всеки рутер в мрежата поддържа маршрутизираща таблица с размер, равен на броя на различните ресурси в мрежата. Всеки елемент в тази таблица е минималното разстояние, измерено в хопове, от този рутер до всички ресурси в мрежата.

Всяка мрежова система се характеризира с това, че ресурсите могат да излязат от мрежата по всяко едно време. Един ефективен механизъм за откриване на ресурси в мрежа трябва да преодолява динамичността на една такава система. В [5] е предложен алгоритъм за откриване на ресурси в динамична мрежа, който е съобразен с този проблем. Предложеният механизъм, наречен премаршрутизиращи таблици, е базиран на механизма маршрутизиращи таблици. Механизмът на премаршрутизиращите таблици може ефективно да се справя със събития, при които ресурсите стават неактивни, както постоянно се случват в динамичните мрежи.

Алгоритъмът на премаршрутизиращите таблици се състои в това, че всеки рутер в мрежата поддържа таблица, в която има информация на колко възела от него се намира най-близкият ресурс от даден тип. По този начин при постъпила заявка за даден ресурс, заявката се изпълнява по възможно най-краткия път. Ако обаче при заявката се установи, че ресурсът е излязъл от мрежата, тогава премаршрутизиращите таблици на всички рутери, които са били свързани именно с дадения ресурс, се изменят така, че да могат да изпълнят заявка за такъв тип ресурс, но на друго място, в което ресурсът е онлайн.

Друг механизъм за откриване на ресурси е така наречения хибридно откриване на ресурси. Хибридният механизъм работи по начин базиран на произволно обхождане. Когато в даден рутер се направи заявка за определен ресурс, той проверява локалните ресурси на този рутер. Ако ресурсът не е намерен там, заявката произволно се изпраща към някой от съседските възли. За да се избегнат нежелани цикли възлите, които са посещавани при дадена заявка, се маркират.

По този начин се избягва повторното преминаване тях. При достигне до ресурс, който е неактивен се налага преизчисляване на таблиците [6].

## РЕАЛИЗАЦИЯ

Проектираното приложение за симулация на откриване на ресурси в динамична мрежа реализира двата описани подхода. Най-голямото предимство на тази реализация е графичния интерфейс, който осигурява доста по-голяма гъвкавост при използването ѝ. Визуализацията на резултатите спомагат за изследването на предимствата на метода с премаршрутизиращите таблици. В графичната програма резултатите веднага излизат под формата на таблици и диаграми, което улеснява работата и дава по-голяма бързина.

Освен всичко друго са добавени много по-големи възможности за задаване на различни настройки като например брой различни видове ресурси, честота на промяна на състоянието (онлайн/офлайн) на ресурсите (фиг. 1).

Едно от основните предимства на предложеното решение се състои в това, че симулира излизането и връщането на ресурсите в мрежата. По такъв начин методът, който е обект на изследване, може да се разглежда в среда, наистина близка до реалната.

С оглед на ефективността на метода беше взето следното решение: когато един ресурс се върне в мрежата, таблиците на рутерите, които са били свързани с него се променят само тогава, когато бъде получена заявка за него в рутера, с който е свързан.

Друг възможен вариант е мрежата да се сканира постоянно и при връщане на ресурс в мрежата винаги да се променят маршрутизиращите таблици на рутерите. Причината за вземане на първото решение е, че ако се избере втория вариант, тогава трафикът в мрежата ще стане наистина тежък и тя няма да работи ефективно.

В повечето случаи е по-добре да се направи по-голям брой хопове до дадено място, отколкото да се правят излишни хопове и манипулации за променяне на таблици на рутери за ресурси, които не се знае дали изобщо ще бъдат искани в мрежата и дали в следващия момент няма пак да станат неактивни и да се налага да се прави обратната процедура.

**Фиг. 1.** Настройки за мрежата ресурсите и методите в приложението

В повечето случаи е по-добре да се направи по-голям брой хопове до дадено място, отколкото да се правят излишни хопове и манипулации за променяне на таблици на рутери за ресурси, които не се знае дали изобщо ще бъдат искани в мрежата и дали в следващия момент няма пак да станат неактивни и да се налага да се прави обратната процедура.

Основните класове в реализацията са за представяне на рутерите и ресурсите.

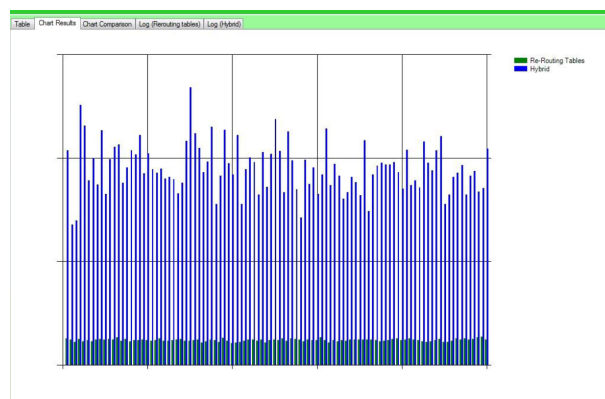
Имплементацията на класа “Network” е свързана и с двата класа и изпълнява всички операции в мрежата. Основната функция SimulateRequests изпълнява заявките в симулацията. Освен това, както по-горе беше споменато, осигурява време, в което ресурсите могат да сменят състоянието си от активно в неактивно и обратно.

Това време е реализирано с таймер System.Threading.Timer, който се изпълнява на отделна нишка, когато нишката, е в режим sleep. Това позволява на потребителя много по-лесно да задава различни настройки за това как и колко често ресурсите да променят състоянието си.

Също така функцията SimulateRequests се изпълнява на различна нишка от тази на главното приложение, което подобрява работата и прави програмата по-бърза на многоядрени процесори и да отговаря на последните стандарти в програмирането - многонишково програмиране. С пускането на самото изпълнение на заявките на отделна нишка, става възможна работата с други функции на приложението, докато заявките се изпълняват.

## РЕЗУЛТАТИ

Наблюденията на резултатите, в реализираният симулационен инструмент за откриване на ресурси в динамична мрежа, се осъществяват таблично (фиг. 2) и графично (фиг. 3).



**Фиг. 2.** Таблично сравняване на резултатите от откриването на ресурси в динамична мрежа

Механизмът на премаршрутизиращите таблици доказва способността си да открива ресурсите за изключително малък брой хопове и с много голяма стабилност.

Увеличаването на броя на ресурсите никак не влияе на резултатите.

| Resource | Re-Routing Tables | Hybrid    | Comparison (%) |
|----------|-------------------|-----------|----------------|
| 7        | 4.400769          | 39.034462 | 88.70507       |
| 8        | 4.830396          | 34.82853  | 86.1046        |
| 9        | 4.954545          | 45.3271   | 89.09305       |
| 10       | 4.817204          | 32.95192  | 85.38111       |
| 11       | 5.009293          | 39.68     | 87.37586       |
| 12       | 4.917204          | 42.17778  | 88.57881       |
| 13       | 5.242362          | 40.62049  | 87.57726       |
| 14       | 4.983333          | 35.17708  | 86.97668       |
| 15       | 4.916867          | 38.19318  | 87.12685       |
| 16       | 4.505155          | 41.45283  | 89.13185       |
| 17       | 4.679612          | 40.62942  | 88.48391       |
| 18       | 4.917204          | 40.62942  | 88.48391       |
| 19       | 4.800091          | 36.95506  | 86.98938       |
| 20       | 4.689321          | 40.82292  | 88.51302       |
| 21       | 4.648148          | 37.82801  | 87.71079       |
| 22       | 4.718814          | 37.05808  | 87.27115       |
| 23       | 5.190393          | 37.91811  | 86.56607       |
| 24       | 4.636364          | 36.30233  | 87.08337       |
| 25       | 4.63              | 36.30275  | 87.24615       |
| 26       | 4.688714          | 35.82353  | 86.92001       |

**Фиг. 3.** Графично сравняване на резултатите от откриването на ресурси в динамична мрежа

От всички симулации с реализираното приложение се вижда, че хибридният механизъм е доста по-нестабилен при малки и средни мрежи, докато при големи е малко по-добър, но повече от два пъти по-слаб от премаршрутизиращите таблици.

## ЗАКЛЮЧЕНИЕ

Реализираното приложение за симулиране на процесите при откриване на ресурсите в динамична мрежа има възможност за предоставяне на събития, при които различни ресурси стават неактивни и не могат да бъдат достъпни. Поддържат се два различни подхода за откриване на ресурси: премаршрутизиращи таблици и хибриден алгоритъм.

Изпълнени са различни тестове в малки, средни и големи по размер мрежи, като за изследване на механизмите за откриване на ресурси. Методът на премаршрутизиращите таблици се представи изключително добре и стабилно във всички тестове и изключително много превъзхожда хибридният, който дава непостоянни и разпокъсани резултати, независимо от размера на мрежата.

## БЛАГОДАРНОСТИ

Работата, представена в тази статия, е финансирана по проект 102pd167-09 на НИС при ТУ-София 2010-2011 г.

## REFERENCE

- [1] Ye Zhu, Junzhou Luo, Teng Ma, Dividing Grid Service Discovery into 2-stage matchmaking, ISPA 2004, LNCS, vol. 3358, 2004, pp. 372–381.
- [2] N.A. Al-Dmour, W.J. Teahan, Peer-to-Peer protocols for resource discovery in the grid, Parallel and Distributed Computing and Networks, Innsbruck Austria, 2005..
- [3] Ye Zhu, Junzhou Luo, Teng Ma, Dividing Grid Service Discovery into 2-stage matchmaking, ISPA 2004, LNCS, vol. 3358, 2004, pp. 372–381.
- [4] Wei Li, Zhiwei Xu, Fangpeng Dong, Jun Zhang, Grid Resource Discovery based on a Routing-Transferring Model, Grid 2002, LNCS, vol. 2536, 2002, pp.145–156.
- [5] Konstantinos I. Karaoglanoglou, Helen D. Karatza, Resource Discovery in a dynamical grid based on Re-routing Tables Simulation Modelling Practice and Theory, Volume 16, Issue 6, July 2008, Pages 704-720
- [6] S. Castano, A. Ferrara, S. Montanelli, G. Racca, Matching techniques for resource discovery in distributed systems using heterogeneous ontology descriptions, in: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04), 2004.