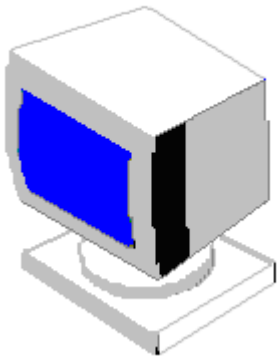
ТЕХНИЧЕСКИ
УНИВЕРСИТЕТ
ВАРНА

КОМПЮТЪРНИ НАУКИ
и
ТЕХНОЛОГИИ

година I брой 1/2003

COMPUTER SCIENCE
AND
TECHNOLOGIES



IEEE
Bulgaria
Communications
Chapter

TU-Varna
FCA
Faculty of Computing & Automation

Компютърни науки и технологии

Издание
на Факултета по изчислителна техника и автоматизация
Технически университет - Варна

Редактор: доц. д-р О. Железов
Гл. редактор: доц. д-р П. Антонов

Печат: ТУ-Варна

За контакти:

Технически университет - Варна
ул. Студентска 1, 9010 Варна, България
тел/факс: (052) 383 320
e-mail: peter.antonov@ieee.org
ognyanz@yahoo.com

ISSN 1312-3335

Computer Science AND Technologies

Publication
of Computing and Automation Faculty
Technical University of Varna

Editor: Assoc. Prof. O. Zhelezov
Chief Editor: Assoc. Prof. P. Antonov

Printing: TU-Varna

For contacts:

Technical University of Varna
1, Studentska Str., 9010 Varna, Bulgaria
Tel/Fax: (+359) 52 383 320
e-mail: peter.antonov@ieee.org
ognyanz@yahoo.com

ISSN 1312-3335

СЪДЪРЖАНИЕ

I Научни статии

CONTENTS

I Scientific Publications

СЪДЪРЖАНИЕ

CONTENTS

Уважаеми читатели,

Пред Вас е първият брой на научно-приложното списание на катедра “Компютърни науки и технологии” (КНТ) в Технически университет – Варна. На страниците на това списание ще намират отражение научни и приложни постижения в областта на информационните технологии на сътрудниците на катедрата и на наши колеги от страната и чужбина. Изданието стартира през настоящата юбилейна 2003 година, в която катедра КНТ отбелязва две значими събития: 35 години от създаването си и 20 години опит в подготовката на компютърни инженери, които се реализират успешно както в страната, така и в чужбина.

Понастоящем катедра КНТ е най-мощната и авторитетна катедра в университета. Тя осъществява общообразователната компютърна подготовка за всички специалности, обучението за придобиване на образователно-квалификационните степени (ОКС) “бакалавър” и “магистър” по специалността “Компютърни системи и технологии” и за получаване на образователна и научна степен “доктор”, както и организирането и провеждането на редица курсове за повишаване на квалификацията на студенти и специалисти. В катедрата са създадени и успешно функционират Мрежова академия на Cisco Systems и Microsoft академия за информационни технологии.

Катедра КНТ поддържа ползотворни контакти със сродни организации и катедри от страната и чужбина, участва в международни проекти на Европейската общност и има договори за сътрудничество и обмен на студенти и преподаватели с Университета Абърти в гр. Дънди (Великобритания), Технически университет в Лисабон (Португалия), Технически университет в Магдебург (Германия) и др.

В катедрата работят 48 щатни сътрудника, от които 1 професор (доктор на техническите науки), 12 доценти и 8 главни асистенти със степен “доктор”, 18 главни асистенти, 3 асистенти, 5 инженери по поддръжка на компютърна техника и 1 секретар. Сътрудниците на катедрата полагат постоянни усилия за усъвършенстване на учебния процес и научно-изследователската дейност.

От учебната 2003/2004 година катедрата провежда обучение за получаване на ОКС “бакалавър” по нов учебен план, съобразен изцяло с изискванията на международните организации IEEE и ACM . За ОКС “магистър” понастоящем се предлагат 4 магистърски програми: Компютърни системи и мрежи, Програмни системи и технологии, Информатика и Microsoft информационни технологии, по които се обучават бакалаври от ТУ Варна и други университети в страната.

Основните направления на научна работа в катедрата са: софтуерно инженерство, програмни технологии в Интернет, разработка на адаптивни компютърни контролери, компютърни мрежи, моделиране и анализ на компютърни системи и мрежи, бази данни и информационни системи, паралелни архитектури и изчисления, мултимедийни системи, програмни технологии в банковата сфера, информационна сигурност, цифрова обработка на сигнали и изображения, специализирани микропроцесорни системи и др.

Успех на новото списание

доц. д-р инж. Петър Антонов
ръководител катедра "Компютърни науки и технологии"
в Технически университет - Варна

ОЦЕНКА НА ВЕРОЯТНОСТТА ЗА НЕСВЪРЗАНОСТ НА КОМУНИКАЦИОННИ И КОМПЮТЪРНИ МРЕЖИ

Петър Ц. Антонов

Резюме: Предлага се обобщен подход и алгоритъм за оценка на вероятността за несвързаност на комуникационни и компютърни мрежи, базиран на аналитични модели и ускорена процедура за симулация, който може да бъде използван за надеждностен анализ на различни топологии на свързване в процеса на мрежово проектиране.

A CALCULATION OF THE DISCONNECTEDNESS PROBABILITY OF COMMUNICATION AND COMPUTER NETWORKS

P. T. Antonov

Abstract. An integrated approach and algorithm for a calculation of the disconnectedness probability of communication and computer networks with various size are presented. The algorithm bases oneself upon analytical models and a fast simulation procedure. It can to use for a reliability analysis of different interconnection topologies in the networks design process.

Увод

Съвременните комуникационни и компютърни мрежи са сложни системи, при изграждането и експлоатацията на които се изразходват значителни ресурси. Това обуславя необходимостта от сериозен подход към етапа на проектиране и анализ на всички съществени характеристики на тези мрежи.

Една от най-важните характеристики на съвременните мрежи е надеждността [1,2,3,4,5,6 и др.]. Високата надеждност на функциониране, дори и в случаите на отказ на съставляващи компоненти, е едно от главните изисквания към изградените мрежи. Провежданите през последните години проучвания показват, че потребителите на комуникационни и компютърни мрежи в развитите страни вече извеждат на първо място по важност показателите за надеждност.

Един от основните надеждностни показатели на мрежите е вероятността за свързаност (или несвързаност) [1,2,3,5 и др.]. Тази вероятност се определя като вероятност за свързано (или несвързано) състояние на конкретната мрежа, като под несвързано състояние се разбира такова състояние, при което е нарушена връзката между поне една двойка кореспондиращи възли на мрежата, в резултат на откази на комуникационни линии и/или други възли.

Оценката на вероятността за несвързаност може да се определи с помощта на аналитични или имитационни модели [1,2,3,5]. На практика, тази оценка обикновено се получава чрез имитационно моделиране с използване на метода Монте Карло.

Класическото прилагане на този метод, обаче, е съпроводено с два съществени недостатъка, влиянието на които нараства с увеличаване на размерността на мрежите: от една страна,

излишно се генерират и анализират топологични подструктури, получени при по-малка кратност на отказите на комуникационни линии и/или възли, в сравнение със стойностите на коефициентите за свързаност на мрежите, а от друга страна, поради малките стойности на вероятностите за отказ на възлите и на комуникационните линии, получаването на оценката на вероятността за несвързаност на мрежите със задоволителна точност изисква значителни разходи на време и машинни ресурси (посочените недостатъци са присъщи и на приведената в [1] програма за оценка на вероятността за несвързаност на компютърни мрежи, което я прави практически неефективна). Решаването на проблемите, свързани с моделирането на малко вероятни събития, каквито са и отказите на комуникационни линии и възли в мрежите, предполага разработването на усъвършенствани имитационни алгоритми за оценка на вероятностно-времевите характеристики на големи мрежи [2,5], в частност и за оценка на вероятността за несвързаност. В [2] е предложен такъв машинен алгоритъм за ускорено имитационно моделиране и оценка на вероятността за несвързаност, при едновременно отчитане на възможните откази на линии и възли. Тази задача е сложна и не се поддава на ефективно аналитично решение. В случай на мрежи с ограничена размерност, обаче, както и при отчитане само на отказите на комуникационни линии в мрежата, необходимостта от имитационно моде-лиране отпада, а вероятността за несвързаност може да се определи по-лесно с помощта на аналитични модели.

В тази връзка, в настоящата статия се разработва обобщен подход и алгоритъм за машинна оценка на вероятността за несвързаност на мрежи с различна размерност, на базата на комбинирано използване на аналитични модели и ускорена процедура за имитационно моделиране. При това, посочената оценка може да се получи с различна желана точност, в съответствие с конкретните изисквания. Алгоритъмът отчита отказите само на комуникационните линии, които са най-вероятните източници на проблеми със свързаността. Освен това, в съвременните мрежи широко се използва резервирането на линии [4,7], с цел осигуряване на по-висока надеждност. В този смисъл предлаганият алгоритъм позволява да се направи и предварителен анализ на различни варианти на такова резервиране.

1. Постановка на задачата

Дадена е мрежа с m кореспондиращи възела и n комуникационни линии, която може да се представи чрез неориентиран граф с m възела и n свързващи дъги, в съответствие с конкретната топологична структура. Освен това, $\{q_i\}$- дискретно разпределение на вероятностите за отказ на комуникационните линии $(i=1 \div n)$. При това, ако между дадена двойка възли в мрежата се използват допълнително r резервни комуникационни линии, то връзката между тези възли в графа може да се представи с една дъга с резултираща надеждност $q_i = \prod_{\alpha=1}^{r+1} q_{i\alpha}$, където $q_{i\alpha}$ - вероятност за отказ на линия α в i-та дъга.

Да се определи оценката Q на вероятността за несвързаност на мрежата, вследствие на откази на комуникационни линии.

2. Решение на задачата

При решението на задачата се разглеждат следните четири основни ситуации: малки и големи мрежи, съответно, с равнонадеждни и с неравно-

надеждни комуникационни линии (в случая, разделянето на мрежите на малки и големи е условно, тъй като в зависимост от конкретните стойности на *m* и *n*, необходимата точност на оценката и достъпните ресурси, анализираната мрежа може да се отнесе към един или друг от посочените класове). При всяка от ситуациите отказите на комуникационните линии се разглеждат като независими случайни събития.

Всяка мрежа в даден момент може да се намира в едно от следните *(n+1)* състояния:

$\Omega_0, \Omega_1, \Omega_2, ..., \Omega_l, ..., \Omega_n,$ където

$\Omega_l -$ състояние на отказ на *i* произволни комуникационни линии.

От своя страна, във всяко от горните състояния мрежата може да е свързана или несвързана, т.е. всяко състояние Ω_l се характеризира с C_n^l на брой различни топологични подструктури (подмрежи), част от които са свързани, а останалите - несвързани.

Нека *Z=1* да съответствува на несвързана мрежа, а *Z=0* - на свързана такава. Тогава, оценката Q на вероят-ността за несвързаност на мрежата може да се определи от формулата:

$$Q = Bep(Z=1) =$$

(1) $$\sum_{l=0}^{n} Bep(\Omega_l) Bep[(Z=1)/\Omega_l]$$

Нека

R - коефициент на дъгова свързаност на мрежата;

h - брой на комуникационните линии (дъги) в свързващото дърво на гра-фа на анализираната мрежа;

$$Bep(\Omega_l) = P_l(n);$$
$$Bep[(Z=1)/\Omega_l] = Q(l).$$

Тъй като, при *l<R* топологичните подструктури на мрежата са свързани, а при *l>(n-h)* - несвързани, то

(2) $$Q = \sum_{l=R}^{n-h} P_l(n) Q(l) + \sum_{l=n-h+1}^{n} P_l(n).$$

Това съотношение се използва като базово за всяка от разглежданите по-долу четири основни ситуации.

А. Малки мрежи с равнонадеждни комуникационни линии

В случая, $\forall i, q_i = q$ и за оценка на Q се анализират напълно всички със-тояния на мрежата $\Omega_l, l = R \div n - h$.

От своя страна,

(3) $P_l(n) = C_n^l q^l (1-q)^{n-l}$ и

(4) $$Q(l) = \frac{T_l}{C_n^l},$$

където T_l - брой на несвързаните топо-логични подструктури в състояние Ω_l.

С отчитане на (3) и (4), съотношение (2) се записва окончателно, като

(5) $$Q = \sum_{l=R}^{n-h} q^l (1-q)^{n-l} T_l + \sum_{l=n-h+1}^{n} C_n^l q^l (1-q)^{n-l}$$

При $n \to \infty$ и $q \to 0$, в съотношение (3) може да се използва формулата на Поасон, т.е.

(6) $$P_l(n) = C_n^l q^l (1-q)^{n-l} = \frac{(qn)^l}{l!} e^{-qn}.$$

На практика, горната замяна е с достатъчна точност при *n>10* и *q<0.1*. С отчитане на (6), формула (2) приема вида:

(7) $Q = \sum_{l=R}^{n-h} \frac{T_l (qn)^l}{C_n^l l!} e^{-qn} + \sum_{l=n-h+1}^{n} \frac{(qn)^l}{l!} e^{-qn}$.

В случаите, когато вторите слагаеми във формулите (5) и (7) имат несъществена значимост за крайните резултати, те могат да се пренебрегнат, т.е.

(8) $Q = \sum_{l=R}^{n-h} q^l (1-q)^{n-l} T_l = \sum_{l=R}^{n-h} \frac{T_l (qn)^l}{C_n^l l!} e^{-qn}$

Блок-схемата на обобщения алгоритъм за изчисляване на Q, в съответствие с (5), е представена на фиг.1, където *G(m,n)* е граф на мрежата, който се описва чрез бинарна матрица на връзките с размерност $(m \times m)$, а

$T[l] = T_l$, $C[n,l] = C_n^l$ и $B[n,l] = q^l (1-q)^{n-l}$.

Точността на определената оценка на Q в случая зависи изцяло от точността на измерване на входния параметър *q*.

Ако за оценка на Q се използва формула (8), то блоковете с номера 9, 10 и 11, в алгоритъма на фиг.1, ще от-паднат. При това, процедурата за нами-ране на свързващото дърво на мрежа-та (блок 3), също може да отпадне, а за *h* да се приеме *h=m-1*, тъй като в свързващото дърво на мрежа с *m* въ-зела не може да има по-малко от *(m-1)* дъги.

В представеният алгоритъм най-трудоемка е процедурата за генериране и анализ на свързаността на топологичните подструктури с *l* отказали комуникационни линии (блок 5). Общият брой *K* на тези подструктури е $K = \sum_{l=R}^{n-h} C_n^l$ и в зависимост от стойността на *K*, конкретната мрежа може да се отнесе към класа на малките или големи мрежи.

B. Малки мрежи с неравнонадеждни комуникационни линии

В случая, с входните данни се въвежда и дискретното разпределение на вероятностите за отказ на комуникационните линии $\{q_i\}$, което може да се запише като *n*-мерен вектор или в надеждностна матрица с размерност $(m \times m)$ и елементи $q_{\beta\gamma}$ - вероятност за отказ на линията между възлите β и γ на мрежата (при отсъствие на линия между два възела, съответният елемент на матрицата е равен на 1).

Съотношението за $P_l(n)$ и Q се записват, съответно, като

(9) $P(n) = \sum_{\eta=1}^{C_n^l} \prod_{j=1}^{l} q_{j\eta}^{(l)} \prod_{j=l}^{n} (1 - q_{j\eta}^{(l)})$ и

(10) $Q = \sum_{l=R}^{n-h} \sum_{\eta=1}^{C_n^l} z_\eta^{(l)} \prod_{j=1}^{l} q_{j\eta}^{(l)} \prod_{j=l}^{n} (1 - q_{j\eta}^{(l)}) + \sum_{l=n-h+1}^{n} P_l(n)$,

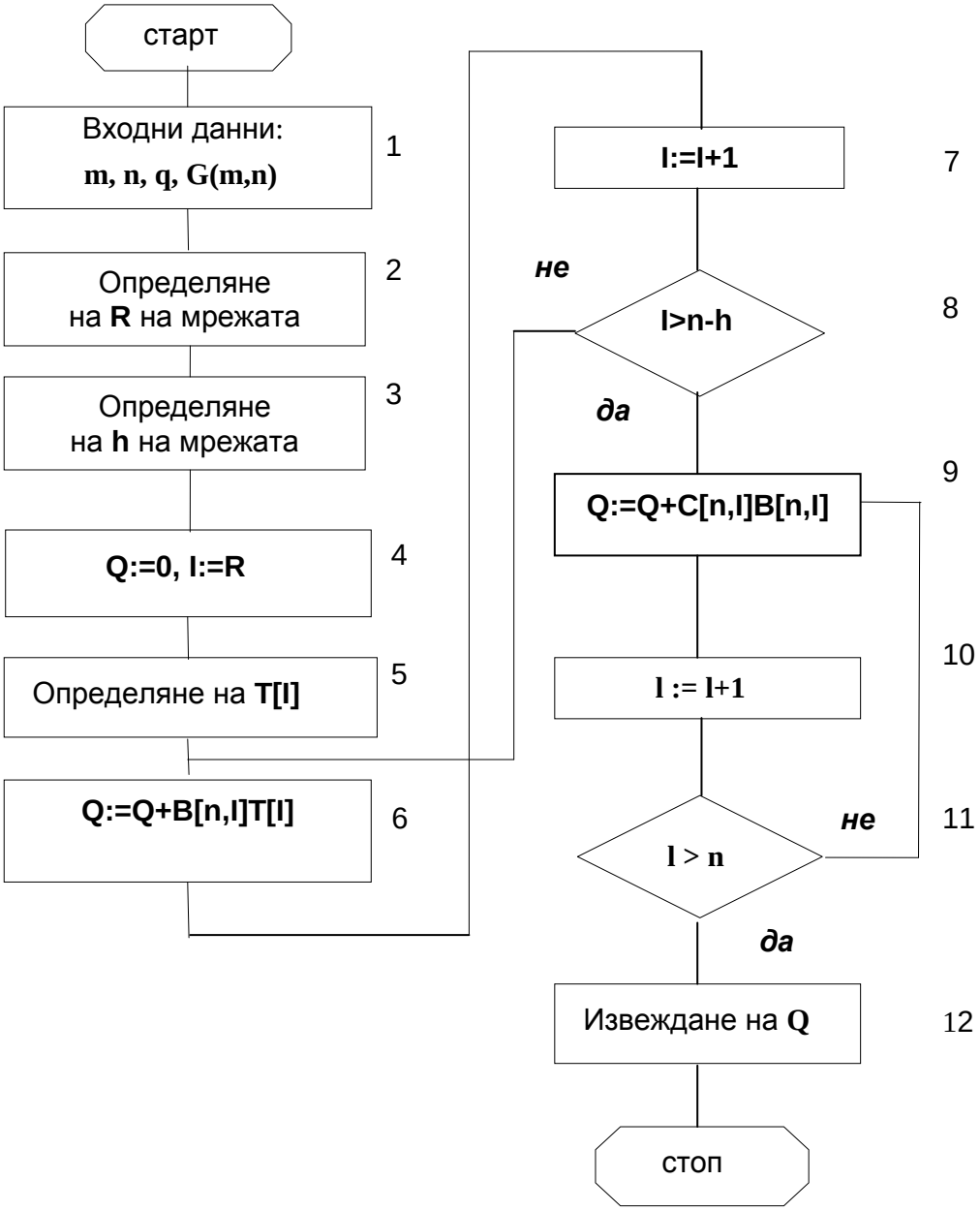
където $z_\eta^{(l)} = 1$, ако η-та топологична подструктура в състояние Ω_l е несвързана и $z_\eta^{(l)} = 0$ - в противоположния слу-чай.

По формула (10) може да се състави алгоритъм за определяне на Q, като точността на оценката ще зависи само от точността на измерване на входните параметри q_i. И в този случай, най-трудоемка е процедурата за генерация и анализ на свързаността на подмрежите с *l* отказали линии.

Ако влиянието на второто слагаемо в (10) е несъществено, то може да се пренебрегне, намирането на свързващото дърво на мрежата - да отпадне,а за *h* - да се приеме *h = m-1*.

C. Големи мрежи с равнонадеждни комуникационни линии

При големи стойности на *K*, намирането на точна оценка на Q за разумно време е практически невъзможно, а и в много случаи не е необходимо. При това положение, за всяко от разглежданите състояния на мрежата може да се проведе имитационно моделиране с използване на метода Монте Карло, като за целта се генери-



Фиг. 1

рат и анализират случайни топологични подструктури (подмрежи), чиито брой N_l е по-малък от C_n^l.

Оценката на Q, в случая, може също да се определи на базата на съотношение (2), $P_l(n)$ - съгласно (6), а $Q(l)$ - Като

(11) $\forall l,\ Q(l) \approx \frac{S_l}{N_l}$, където

S_l - брой на генерираните несвързани подмрежи с *l* отказа на комуникационни линии;

N_l - общ брой на генерираните в машинния експеримент подмрежи с *l* отказа на линии за състояние Ω_l.

Очевидно е, че в този случай точността на оценките на $Q(l)$ и Q ще зависи съществено от N_l.

Ако грешките в тези оценки обозначим съответно чрез $|\Delta Q(l)|$ и $|\Delta Q|$, то в най-неблагоприятния случай

(12) $$|\Delta Q| = \sum_{l=0}^{n} P_l(n)|\Delta Q(l)| = \sum_{l=R}^{n-h} P_l(n)|\Delta Q(l)|.$$

Вижда се, че за намаляване на $|\Delta Q|$, оценките $Q(l)$ за по-вероятните състояния на мрежата $\boldsymbol{\Omega_l}$, следва да се определят с по-голяма точност, т.е.

(13) $$|\Delta Q(l)| = \frac{|\Delta S_l|}{N_l} \sim \frac{1}{P_l(n)} \text{ и } N_l \sim P_l(n).$$

Следователно, ако *N* е необходим брой статистически опити при класическото използване на метода Монте Карло, то

(14) $$N_l = P_l(n)N \quad \text{и} \quad Q(l) = \frac{S_l}{P_l(n)N}.$$

Тогава

(15) $$Q = \frac{1}{N}\sum_{l=R}^{n-h} S_l + \sum_{l=n-h+1}^{n} C_n^l q^l (1-q)^{n-l} \quad \text{или}$$

(16) $$Q = \frac{1}{N}\sum_{l=R}^{n-h} S_l + \sum_{l=n-h+1}^{n} \frac{(qn)^l}{l!} e^{-qn}.$$

Обобщеният алгоритъм за намиране на оценката на *Q* за разглеждания случай е представен на фиг.2, където

$S[l] = S_l$, $P[n,l] = P_l(n)$, $N[l] = N_l$,

$X[\eta,l] = X_\eta^{(l)}$, $z[\eta,l] = z_\eta^{(l)}$

$X_\eta^{(l)} = \{x_{1\eta}^{(l)}, x_{2\eta}^{(l)}, ..., x_{i\eta}^{(l)}, ..., x_{n\eta}^{(l)}\}$- вектор на състоянията на комуникационните линии, като $x_{i\eta}^{(l)} = 1$ съответства на отказ на *i*-та линия в η-та генерация в състояние Ω_l, а $x_{i\eta}^{(l)} = 0$ - на работоспособна такава;

$z_\eta^{(l)}$- показател за свързаност на η-та подмрежа с *l* отказали линии ($z_\eta^{(l)} = 1$ - съответства на несвързана подмрежа, а $z_\eta^{(l)} = 0$ - на свързана такава).

За генерацията на векторите $X_\eta^{(l)}$ е необходимо да се определят по случаен начин номерата на *l* отказали линии, което може да се реализира с помощта на датчик за равномерно-разпределени случайни числа в интервала *(1,n)* [възможно е използването и на датчик за равномерно-разпределени случайни числа в интервала *(0,1)* и претеглената вероятност $q^* = 1/n$].

В представеният алгоритъм на фиг. 2 реалният брой на статистическите опити N_R е по-малък от N, тъй като

(17) $$N_R = N - N\sum_{l=0}^{R-1} P_l(n) - N\sum_{l=n-h+1}^{n} P_l(n) = N\sum_{l=R}^{n-h} P_l(n).$$

При това, разликата $(N - N_R)$ нараства с увеличаване на стойността на коефициента на свързаност на мрежата *R*.

Например, ако мрежата е с параметри *n=10*, *q=0.1*, *R=2* и N=1000, то математическото очакване $M(N_{out})$ на броя на излишните статистически опити

N_{out} при класическото използване на метода Монте Карло, ще бъде $M(N_{out}) > P_{l=0}(10)N + P_{l=1}(10)N = 728$. При *n=20*, *q=0.1*, *R=2* и *N=10000*, $M(N_{out}) > 3918$, а ако при същите други параметри *R=3*, то $M(N_{out}) > 6870$.

Представеният алгоритъм на фиг.2 не реализира излишни статистически опити и освен това дисперсията на оценката на *Q* се получава по-малка, тъй като се анализират, пропорционално на тяхните вероятности, всички множества (състояния) от топологични подструктури на мрежата с *l* отказали линии.

Ако вероятностите $P_l(n)$, за *l>n-h*, са достатъчно малки величини, то вторите слагаеми в (15**)** и (16) могат да се пренебрегнат. Тогава, *h=m-1* и блоковете с номера 16, 17 и 18 ще отпаднат от алгоритъма.

D. Големи мрежи с неравнонадеждни комуникационни линии

Тази ситуация се характеризира с различни вероятности за отказ на комуникационните линии на мрежата и голяма стойност на *K*, изискваща неразумен разход на машинно време за точна оценка на *Q*. Поради това, в случая може да се използват предидущите разсъждения, както и обобщения алгоритъм от фиг. 2, но със следните изменения:

а) вместо *q* се въвежда $\{q_i\}$;

б)вероятностите $P_l(n)$ се определят в съответствие с (9);

в) векторите $X_\eta^{(l)}$ се генерират с помощта на датчик за равномерно-разпределени случайни числа в интервала *(0,1)* в съответствие с претеглените вероятности за отказ $\{q_i^*\}$ на комуникационните линии, където

$$(18)\ \forall i, q_i^* = \frac{q_i}{\sum_{i=1}^{n} q_i}, \quad \sum_{i=1}^{n} q_i^* = 1.$$

Тогава

$$(19)\ Q = \frac{1}{N}\sum_{l=R}^{n-h} S_l + \sum_{l=n-h+1}^{n}\sum_{\eta=1}^{C_n^l}\prod_{j=1}^{l} q_{j\eta}^{(l)}\prod_{j=l}^{n}(1-q_{j\eta}^{(l)})$$

С увеличаване на размерността на мрежата значимостта на второто слагаемо в (19) намалява, поради което за *Q* може да се приеме $Q \approx \frac{1}{N}\sum_{l=R}^{n-h} S_l$, където *h=m-1*. Освен това, възможна е допълнителна модификация на алгоритъма за този случай, при което вместо *N* се въвеждат стойностите N_l за всяко състояние на мрежата $\Omega_l, l = R \div n-h$, като с това се избягва трудоемкото изчисляване на вероятностите $P_l(n)$ в съответствие с (9). При това положение вероятността *Q* се определя от съотношението

$$(20)\ Q \approx \sum_{l=R}^{n-h}\sum_{\eta=1}^{N_l} z_\eta^{(l)} V_\eta^{(l)}(n),$$

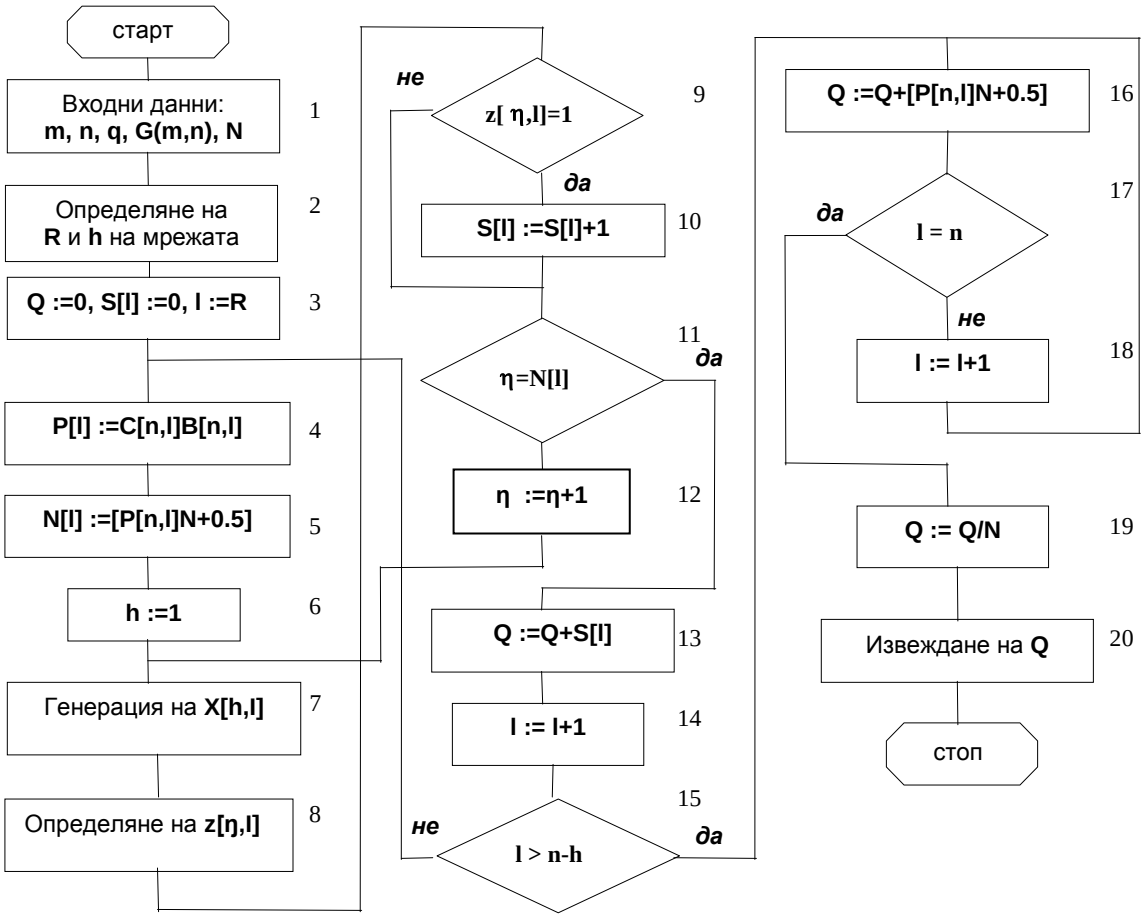
$$V_\eta^{(l)}(n) = \prod_{j=1}^{l} q_{j\eta}^{(l)}\prod_{j=l}^{n}(1-q_{j\eta}^{(l)}),$$

като вероятностите $V_\eta^{(l)}(n)$ се изчисляват само за генерираните несвързани подмрежи, т.е. при $z_\eta^{(l)} = 1$.

Заключение

Разгледаните по-горе обобщен подход и алгоритъм за оценка на несвързаността на комуникационни и компютърни мрежи позволяват създаването на машинна програма, обхващаща възможните ситуации и изчислителни процедури за всяка анализирана мрежа. При това е целесъобразно да се предвиди възможност за диалог и избор на конкретна ситуация и

изчислителна процедура, както и прекратяване на итерационните изчислителни процедури до определена стойност на *l*, ако в дадената ситуация влиянието на състоянията Ω_l на мрежата върху оценката на Q, при по-големи *l*, е несъществено. Такава програма може да се използва, както за определяне на *Q* на дадени мрежи, така и за сравнителен надеждностен анализ на различни топологични структури в процесса на тяхното проектиране. При



Фиг.2

това могат да се анализират и различни варианти на въвеждане на резервни комуникационни линии в мрежите.

Литература

[1].Таненбаум, Е. Мрежи от електронно-изчислителни машини. Прев. от англ. – С.: Техника, 1985.

[2].Кутузов, О., П. Антонов. Подход за ускорено имитационно моделиране и оценка на свързаността на комуникационни мрежи. – Proc. of the Intern. Conf. on Energy and Inform. Systems and Technologies. – Bitola (Macedonia), 2001.

[3].Бертсекас, Д., Р. Галлагер. Сети передачи данных.: Пер. с англ. - М.: Мир, 1989.

[4].Ретана, А. и др. Принципы проектирования корпоративных IP – сетей.: Пер. с англ. – М.: Вильямс, 2002.

[5].Кутузов, О. Методы и модели ускоренной имитации в задачах разработки сетей интегрального обслуживания АСУ. – Диссертация на соискание ученой степени доктора технических наук. – СПб.: СПГЭТУ, 1996.

[6].Ray, K. Cooperative Management of Enterprise Networks. – NY: Kluwer Academic, 2000.

[7].Halsall, F. Multimedia Communications. Applications, Networks, Protocols and Standards. – England: PEL, 2001.

За контакти: доц.д-р инж. П. Антонов
e-mail: peter.antonov@ieee.org

Изчисление на матрици за преход и фалит от исторически данни за рейтинг

Анатолий Ст. Антонов
Снежина К. Петрова

Резюме: В доклада се разглеждат накратко получаването и обработката на вътрешните матрици за преход, изчислени по исторически данни за рейтинг с помощта на Credit Risk Data Supporter, включен в пакета Risk Evaluator.

Computation of transition matrixes from historical ratings

Anatoliy St. Antonov
Snezhina K. Petrova

Abstract: The following document considers the computation and manipulation of the internal transition matrixes from historical ratings with the help of Credit Risk Data Supporter, included in Risk Evaluator package.

1.Увод

Един от най-важните рискови фактори при депонирането на собствения капитал за кредитния риск това е вероятността за фалит (PD - Probability OF Default) на кредитоискателя. Тази оценка трябва да се базира на собствени исрорически данни на банката за различни периоди.

Един от възможните начини на работа е определянето на вероятността за фалит с помощта на вътрешни матрици на преход. Вероятността за фалит се получава в колона D на реда, означен със настоящия рейтинг на кредитоискателя в съответната матрица на преход (колона D в Табл.1). От тук следва задачата за получавнето на собствените матрици на преход с цел преди всичко статистическа обработка на историческите данните.

Постановката на задачата може да бъде следната:

За определен период от време, банката систематично, през фиксирани интервали събира данни и определя рейтинга на представителна извадка от кредитоискатели.

Примерни общи условия:

Честота на определяне на рейтинга – ежемесечно или през три месеца

Представителна извадка – 1000 кредитоискатели

Исторически срок – 3 години

Рейтингът се определя с помощта на обективни рейтинг модели или рейтинг системи, като винаги се използва една и съща рейтингова степенна скала.

Събраните в банката данни могат да се обработват статистически и да се получат коефициентите за фалит и преход за различни рискови периоди. От тези коефициенти може да се получи вероятността за фалит и преход в собствената матрица за преход за период от 1 година.

Процесът повишава качеството на матрицата за преход, получена от историческите рейтинги, тък като освен историческите фалити се вземат под

внимание и историческите данни за преходи на рейтинга за стандартизирани рискови хоризонти. Процесът предполага методи за трансформация на рисковите хоризонти за матрицата за преход (част 3).

2. Рискови хоризонти при матриците за преход.

Решението на поставената задача изисква математически средства за получаването на матриците за преход за по-къси рискови хоризонти (интерполация) или за по-дълги рискови хоризонти (екстраполация) от дадената матрица за преход. Важна подзадача се явява получаването на матрицата за преход с рисков хоризонт 1 месец от матрицата за 1 година и съответно получаването на матрицата за преход за 1 година от дадената матрица за 1 месец.

2.1 Интерполация на вероятностите за преход и за фалит

Вероятността за фалит за по-къси периоди може да се получи чрез експоненциална интерполация на вероятностите за преход. Изхождайки от функцията Hazard Rate[1]:

$$h(t) = \frac{f(t)}{1 - F(t)} = -\frac{S'(t)}{S(t)}$$

приемайки, че условията за фалит са едни и същи за целият период от една година, вероятността за преход и вероятността за фалит за по-къси периоди могат да представят по следният начин:

$$S^{Rating}(t) = 1 - PD^{Rating}(t)$$

$$S^{Rating}(t) = S^{Rating}(1\ Year)^t$$

$$PD^{Rating}(t) = 1 - (1 - PD^{Rating}(1\ Year))^t$$

където
$S^{Rating}(t)$ - вероятността рейтинговата степен да не фалира за време t

$PD^{Rating}(t)$ - вероятността за фалит на рейтинговата степен за време t

Пример:

ако PD(Rating=BBB, 1 година) = 6.8789%,

тогава PD(Rating=BBB, 1 месец) = 0.5922% и

PD(Rating=BBB, 1 ден) = 0.0195%

Експоненциалната интерполация на вероятността за фалит за периоди по-малки от 1 година се извежда на базата на връзката между кумулативната вероятност за фалит и вероятността за нефалит за същия период.

Примерното преобразуване на матрица за преход за 1 година в матрица за преход за 3 месеца е представено в Табл.1.

2.2 Екстраполация на вероятностите за преход и за фалит.

Вероятността за преход и вероятността за фалит за по-дълги периоди могат да се получат чрез експоненциална екстраполация на вероятностите за преход. Формулата се преобразува като:

$$PD^{Rating}(1\ Year) = 1 - (1 - PD^{Rating}(t))^{\frac{1}{t}}$$

Изчислението на матрицата на преходите за 1 година от матрицата на преходите за 3 месеца изисква вдигане на 4/та степен, при 1 месец на 12/та степен. Този подход би довел обаче до грешки, тай като поведението за по-къси периоди се реплицира многократно в рамките на по-дългия период. По тази причина използването на екстраполация е неподходящо, по-подходящо е пълното матрично степенуване на цяла степен.

$$|M(n * t)| = |M(t)|^n \qquad n = 2,3,4...$$

Матрица за преход SnP с 8 рейтингови степени (рисков хоризонт=1 година)								
[%]	AAA	AA	A	BBB	BB	B	CCC	D
AAA	92,6106	6,8418	0,4134	0,1033	0,0309	0,0000	0,0000	0,0000
AA	0,9569	92,6069	5,6711	0,5787	0,0413	0,1175	0,0172	0,0104
A	0,0596	2,4150	91,3435	5,4224	0,4974	0,2203	0,0069	0,0349
BBB	0,0356	0,2739	5,2018	88,1571	5,0454	0,9089	0,1493	0,2280
BB	0,0472	0,1245	0,5203	8,3175	81,9439	7,0225	1,0683	0,9558
B	0,0000	0,0899	0,3050	0,5562	4,3316	80,6401	5,9869	8,0903
CCC	0,1620	0,0000	0,3242	1,2969	1,7832	10,5953	61,7301	24,1083
Интерполирана матрица за преход SnP с 8 рейтингови степени (рисков хоризонт=3 месеца)								
[%]	AAA	AA	A	BBB	BB	B	CCC	D
AAA	98,1068	1,7562	0,1035	0,0258	0,0077	0,0000	0,0000	0,0000
AA	0,2401	98,1193	1,4490	0,1450	0,0103	0,0294	0,0043	0,0026
A	0,0149	0,6093	97,8016	1,3841	0,1246	0,0551	0,0017	0,0087
BBB	0,0089	0,0685	1,3266	96,9876	1,2859	0,2280	0,0373	0,0570
BB	0,0118	0,0311	0,1303	2,1476	95,3673	1,8038	0,2682	0,2398
B	0,0000	0,0225	0,0763	0,1393	1,1009	95,0423	1,5316	2,0870
CCC	0,0405	0,0000	0,0812	0,3258	0,4488	2,7611	89,6785	6,6641

Табл.1 Матрици за преход за 1 година и за 3 месеца

При повдигане на втора степен всяка вероятност за фалит или преход се изчислява като сума от условните преходи към всяка друга рейтингова степен.

Например:

За P(AA ->A) = P(AA- >AAA)*P(AAA ->A) + P(AA ->AA)*P(AA ->A) +...

Резултатите от повдигането на на дадената матрица за преход за три месеца на втора степен (получената матрица за преход е за 6 месеца) и на четвърта степен (получената матрица за за преход е за 1 година) са показани в Табл2.

3. Изчисляване на матрици от исторически рейтинги.

3.1 Изчислителни методи

Формата за изчисляване на вероятностите за преход и фалит от исторически рейтинги (Фиг.2) се извиква с бутона Calculate от формата за матрицата за преход (Фиг.1) от пакета Credit Risk Data Supporter.

Във формата от фиг. 1. са избрани степенната скала за рейтинга и рейтинговата агенция, които служат като критерий за избор на зарежданите от базата данни исторически рейтинги. Потребителят въвежда разглеждания исторически период (Start и End Date), и стартира изчислителния процес с бутона Calculate от втората форма.

Изчисленията се извършват както следва:

- Всички рейтингови данни за избраната рейтингова агенция и рейтингова скала за определеният период се извличат от таблицата CM_PART и се подреждат според ID на участника.

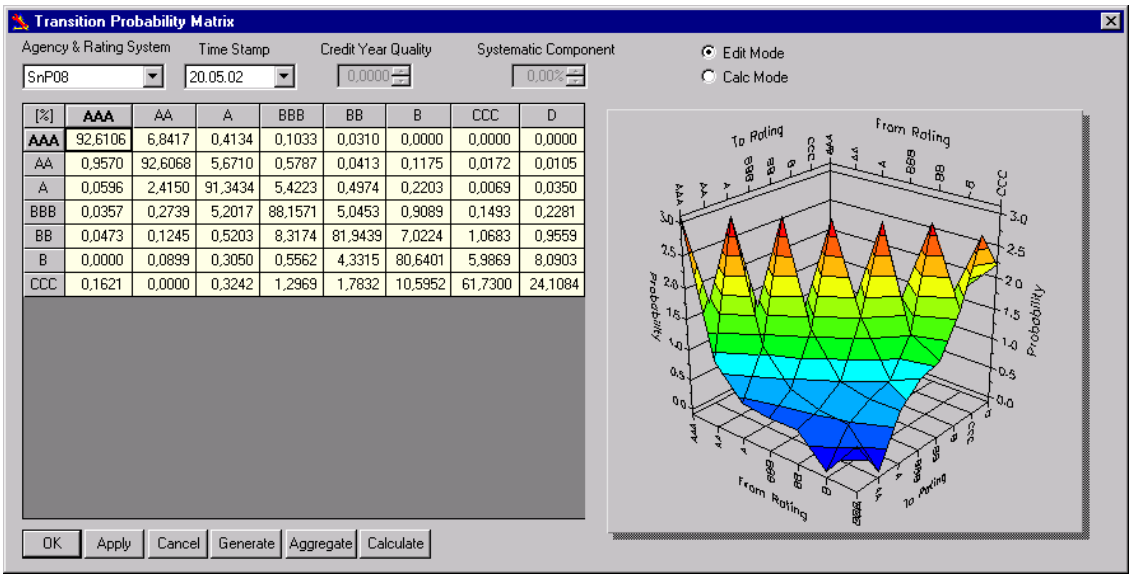
Матрица за преход SnP с 8 рейтингови степени (рисков хоризонт = 3 месеца)								
[%]	AAA	AA	A	BBB	BB	B	CCC	D
AAA	98,1068	1,7562	0,1035	0,0258	0,0077	0,0000	0,0000	0,0000
AA	0,2401	98,1193	1,4490	0,1450	0,0103	0,0294	0,0043	0,0026
A	0,0149	0,6093	97,8016	1,3841	0,1246	0,0551	0,0017	0,0087
BBB	0,0089	0,0685	1,3266	96,9876	1,2859	0,2280	0,0373	0,0570
BB	0,0118	0,0311	0,1303	2,1476	95,3673	1,8038	0,2682	0,2398
B	0,0000	0,0225	0,0763	0,1393	1,1009	95,0423	1,5316	2,0870
CCC	0,0405	0,0000	0,0812	0,3258	0,4488	2,7611	89,6785	6,6641
Матрица за преход SnP с 8 рейтингови степени (рисков хоризонт = 6 месеца)								
[%]	AAA	AA	A	BBB	BB	B	CCC	D
AAA	96,2537	3,4468	0,2286	0,0545	0,0155	0,0008	0,0001	0,0001
AA	0,4714	96,2871	2,8411	0,3033	0,0240	0,0582	0,0086	0,0063
A	0,0308	1,1950	95,6789	2,6997	0,2592	0,1119	0,0049	0,0196
BBB	0,0179	0,1423	2,5870	94,1125	2,4778	0,4628	0,0766	0,1227
BB	0,0232	0,0631	0,2822	4,1363	90,9981	3,4470	0,5247	0,5252
B	0,0008	0,0444	0,1520	0,2972	2,1050	90,3929	2,8322	4,1753
CCC	0,0761	0,0022	0,1593	0,6228	0,8652	5,1092	80,4659	12,6993
Матрица за преход SnP с 8 рейтингови степени (рисков хоризонт = 1 година)								
[%]	AAA	AA	A	BBB	BB	B	CCC	D
AAA	92,6640	6,6393	0,5381	0,1210	0,0319	0,0045	0,0006	0,0006
AA	0,9085	92,7628	5,4631	0,6557	0,0611	0,1145	0,0174	0,0169
A	0,0653	2,2991	91,6494	5,1386	0,5534	0,2306	0,0153	0,0483
BBB	0,0362	0,3043	4,9217	88,7462	4,6039	0,9462	0,1600	0,2808
BB	0,0450	0,1297	0,6418	7,6780	82,9868	6,2988	1,0005	1,2189
B	0,0045	0,0865	0,3022	0,6573	3,8505	81,9276	4,8503	8,3206
CCC	0,1350	0,0121	0,3071	1,1426	1,6069	8,7624	64,8974	23,1365

Табл.2 Матрици за преход за 3, 6 и 12 месеца

- Подредените данни се изследват с ежемесечна честота за разглеждания период, при което се анализират статистически четири рискови хоризонта: преходите за 1 месец, 3 месеца, 6 месеца и 1 година.
- За всеки рисков хоризонт по дължина на времевата ос се пребройват преходите от една рейтинова степен в друга. В карайна сметка тази статистика дава отделните матрици за преход за четирите рискови хоризонта.
- Матриците за преход за 1, 3, и 6 месеца, съгласно част 2.2 ще се повдигнат съответно на 12, 4, и 2-ра степен, за да се получат матрици за преход за една година.
- От четирите матрици за преход за една година се изчислява средна резултатна матрица. Тя може да се съхрани в базата данни като актуална матрица за съответната рейтингова агенция.

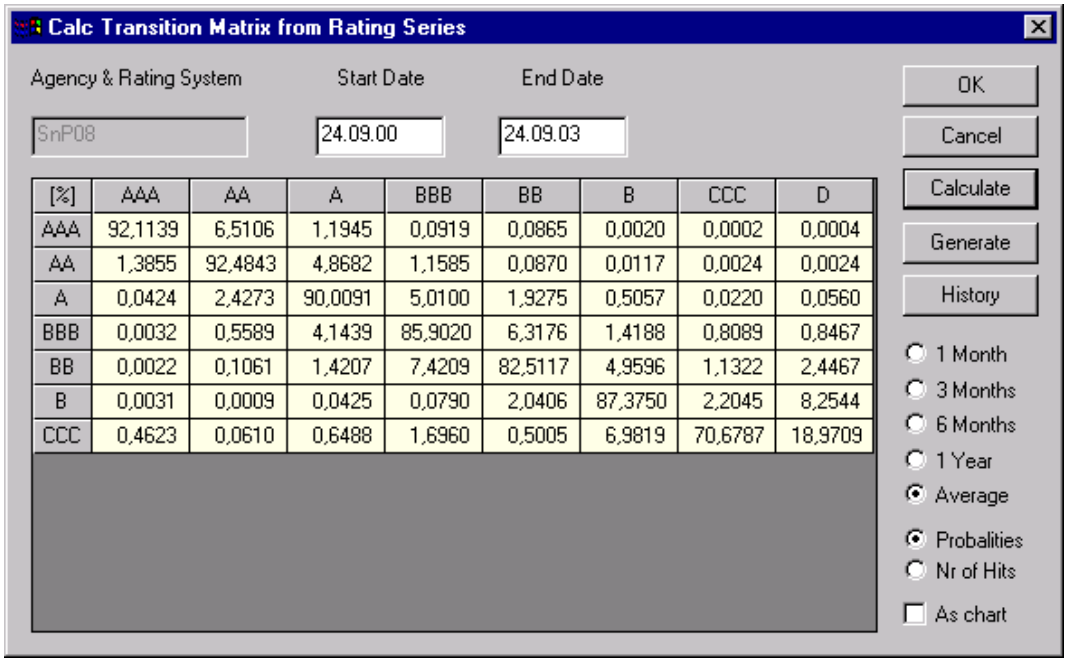
Описаният процес се отличава със следните качества:

- Отчитатат се не само коефициенти за фалит, но и коефициенти на преход, т. е. използва се всичката налична историческа информация, което съответно води до по-голяма статистическа изразителност.
- Процесът позволява изчисление на матриците за преход от собствени исторически рейтингови последователности.
- Статистическата обработка на рейтинговите последователности се извършва за четири стандартизирани рискови хоризонта.



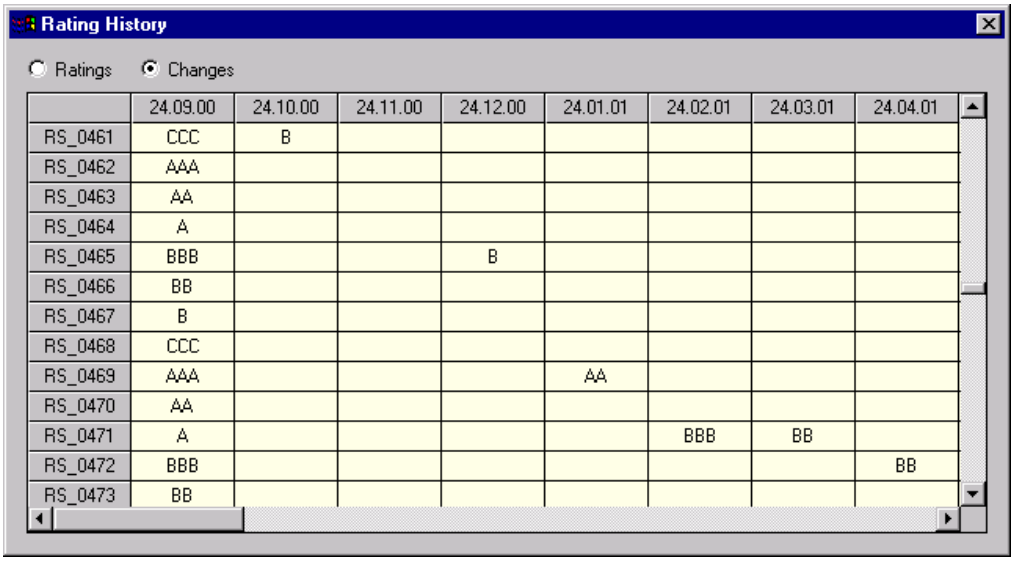
[%]	AAA	AA	A	BBB	BB	B	CCC	D
AAA	92,6106	6,8417	0,4134	0,1033	0,0310	0,0000	0,0000	0,0000
AA	0,9570	92,6068	5,6710	0,5787	0,0413	0,1175	0,0172	0,0105
A	0,0596	2,4150	91,3434	5,4223	0,4974	0,2203	0,0069	0,0350
BBB	0,0357	0,2739	5,2017	88,1571	5,0453	0,9089	0,1493	0,2281
BB	0,0473	0,1245	0,5203	8,3174	81,9439	7,0224	1,0683	0,9559
B	0,0000	0,0899	0,3050	0,5562	4,3315	80,6401	5,9869	8,0903
CCC	0,1621	0,0000	0,3242	1,2969	1,7832	10,5952	61,7300	24,1084

Фиг.1 Форма за матрица за преход в Credit Risk Supporter



[%]	AAA	AA	A	BBB	BB	B	CCC	D
AAA	92,1139	6,5106	1,1945	0,0919	0,0865	0,0020	0,0002	0,0004
AA	1,3855	92,4843	4,8682	1,1585	0,0870	0,0117	0,0024	0,0024
A	0,0424	2,4273	90,0091	5,0100	1,9275	0,5057	0,0220	0,0560
BBB	0,0032	0,5589	4,1439	85,9020	6,3176	1,4188	0,8089	0,8467
BB	0,0022	0,1061	1,4207	7,4209	82,5117	4,9596	1,1322	2,4467
B	0,0031	0,0009	0,0425	0,0790	2,0406	87,3750	2,2045	8,2544
CCC	0,4623	0,0610	0,6488	1,6960	0,5005	6,9819	70,6787	18,9709

Фиг.2 Изчисление на матрицата за преход от исторически рейтинги.



	24.09.00	24.10.00	24.11.00	24.12.00	24.01.01	24.02.01	24.03.01	24.04.01
RS_0461	CCC	B						
RS_0462	AAA							
RS_0463	AA							
RS_0464	A							
RS_0465	BBB			B				
RS_0466	BB							
RS_0467	B							
RS_0468	CCC							
RS_0469	AAA				AA			
RS_0470	AA							
RS_0471	A					BBB	BB	
RS_0472	BBB							BB
RS_0473	BB							

Фиг.3 Представяне на промените на рейтинга.

3.2 Представяне на историческите рейтинги.

Представянето на историческите рейтинги заема своето място след изчислението на матрицата за преход. Данните се сканират и анализират ежемесечно през целия разглеждан период за всеки кредитоискател и се представят таблично. Фиг. 3 показва промените на рейтинга.

Литература:

[1] David X. Li On Default Correlation: A Copula Function Approach, February 2000

Characterization of Default by Time-Until-Default

RiskMetrics Group, Working Paper Number 99-07

[2] Dr. Barry Belkin A one-parameter representation of credit risk and transition matrices RiskMetrics Group, CreditMetrics® Monitor, Third Quarter 1998

Евристични алгоритми за търсене и клъстеризация на софтуер

Виолета Т. Божикова

Резюме: Евристичните алгоритми за търсене се използват широко при решаване на много инженерни проблеми. Те са приложими за задачи, при които добрите решения лесно се познават, но трудно се генерират поради конкуриращи се ограничения, голяма област за търсене и сложна целева функция. Днес, базираната на търсене клъстеризация на софтуер, е един установен подход за клъстеризация, с обещаващи резултати като качество и ефективност. В статията се коментира SOAD - разработен от автора евристичен алгоритъм за клъстеризация.

Heuristic Search algorithms and Software Clustering

Violeta T. Bojikova

Abstract: Heuristic search algorithms have found wide application in engineering problems. Such techniques are applicable when good solutions are easy to recognize but hard to generate, because the competing constraints, the size of the search space and the complexity of the cost function. Search-based software clustering is now an established approach to software clustering, with promising results in terms of quality and efficiency. In this paper, the author comments his software-clustering algorithm SOAD, as a heuristic search algorithm.

1. Въведение

Алгоритми за клъстеризация [2,3,4] се използват в различни области за поддържане процеса на групирането на подобни обекти на системата. Ключова идея на клъстеризацията е да се групират подобни елементи в групировки (клъстери), така че вътре-клъстерното подобие или кохезия да бъдат високи и между-клъстерното подобие или свързаност – ниски.

Алгоритмите за клъстеризация се използват в различни етапи на софтуерния жизнен цикъл. Те се прилагат в процеса на проектиране – за осъществяване на ефективно деление (декомпозиция) на софтуерната архитектура, за валидизиране на декомпозиционни решения.

Тези алгоритми решават основния реинженерингов проблем "Възстановяване на архитектурата", тоест намиране на компонентите и връзките на софтуерната архитектура.

За големи системи, желаните компоненти са абстрактни ("високо ниво") и могат да бъдат моделирани чрез архитектурни продукти – подсистеми. Те не се виждат директно в изходния код на софтуерната система и в условията на неадекватна или липсваща документация се разработват алгоритми, способни да намерят разумна апроксимация на софтуерната архитектура.

"Възстановената" архитектура обезпечава особено актуалните днес реинженерингови задачи:

- пренос на софтуера на нова платформа;
- идентифициране на кандидати за повторно използване ("reuse"), което означава кандидати за софтуерни компоненти - модули,

подсистеми, проектни решения, документация и др. за други програми;

- изучаване на програмата;
- подобряване на софтуерната поддръжка;
- увеличаване надеждността на системата и др.

2. Евристични алгоритми за търсене - изисквания към проблемната област

Евристичните алгоритми за търсене [2,3,4] широко се прилагат за решаване на различни оптимизационни задачи, при наличие на много локални екстремуми, с много параметри и конфликтни ограничения. Те успешно намират приемливи апроксимации на много определени като “NP-пълни” и “NP-сложни” проблеми, при които е невъзможно да се приложат точни аналитични алгоритми, които произвеждат оптималното решение, но е възможно да се определи кой от два кандидата е по-добър.

Евристичните алгоритми се съставят и реализират лесно. Единственият недостатък е че решението може да се окаже далеч от оптималното. Но, това, в редица практически случаи, е за предпочитане, пред алтернативата, за едно безкрайно и безперспективно претърсване за оптималното решение.

Ключът за успешното прилагане на такъв алгоритъм е, преди всичко възможността, за формулиране на проблема, като “проблем за търсене|оптимизация”:

- Голяма област от решения
- Не съществуване на ефикасно и пълно решение.
- Съществуване на подходяща целева функция за оценка на решенията
- Лесно (не скъпо) генериране на начални кандидати за решения (за разумно време).

Градиентният алгоритъм на спускане|изкачване (hill climbing) е най-известният и прилаган евристичен алгоритъм за търсене. При него процесът на търсене типично започва от “случайно” <начално състояние> (тоест, от случайно решение). Оценява се множеството “съседни” състояния (решения) и се избира “подходящо <съседно състояние>”, което става <текущо състояние>, след което процесът се повтаря.

Доминира мнението [2], че преди разработката на каквито и да е други евристични (Tabu Search, Simulated Annealing, Min-Conflicts) или вероятностни алгоритми за клъстеризация (ГА) за оптимизационен проблем, трябва да се тръгне от градиентен алгоритъм (hill climbing), и то – най-простия. Мотивацията е, че ако един такъв алгоритъм (hill climbing) дава достатъчно добри резултати, то е ненужно усилието за разработка на други алгоритми за решавания проблем, например вероятностни алгоритми за търсене (в частност, генетични). Нещо повече, ако резултатите от градиентния алгоритъм не са по-добри от тези на вероятностните алгоритми, то това трябва да се счита за индикация, че или не е разбран проблема, или не е адекватна направената формализация.

3. Проблемът “клъстеризация на софтуер” – “проблем за търсене|оптимизация”

Проблемът “клъстеризация на софтуер” може да бъде видян като “проблем за търсене|оптимизация”:

- Процесът на клъстеризация на софтуер се характеризира с голям брой конкуриращи се и взаимо свързани ограничения.
- Областта от възможни решения може да бъде изключително голяма, което прави изпълнението на оптимален алгоритъм (метод на

"пълното изброяване") невъзможно скъпо (NP-hard).
- Независимо, че не е оправдано или не е твърде ясно, как може да се намери оптималното решение, то относително лесно може да се определи, дали едно решение е по-силно свързано от друго, тоест налице е подходяща целева функция за оценка на решенията.

Съществуват различни алгоритми за разделяне структурата на софтуерната система в подсистеми (клъстери). Те определят клъстери на база на различни критерии: подобие между компонентите, на концептуален анализ, на различни метрики или на информация за изпълнението на системата. Тези алгоритми за клъстеризация дават добри резултати за конкретни типове софтуерни системи [2,3,4]. Чрез серия от публикации, Michell и Mancoridis показват, че базираните на търсене евристични алгоритми (SAHC, NANC) са подходящи за клъстеризация на доста по-голямо разнообразие от системи [3,4].

4. SOAD – суб-оптимален алгоритъм за клъстеризация на софтуер

SOAD е евристичен алгоритъм за клъстеризация (декомпозиция) на софтуер, по критерий за клъстеризация - "минимална външна свързаност", чиято основна задача е намиране на балансирани по тегло клъстери – кандидати за подсистеми, при горен праг W_0 за теглото W_d на клъстерите (тоест, $W_d <= W_0$, $d \in 1..M$, където M-брой на клъстерите).

Основно предположение лежащо в основата на SOAD (и на други алгоритми [3,4], базирани на критерия за клъстеризация "минимална външна свързаност") е че:
- Добре проектираните софтуерни системи са организирани в клъстери с висока кохезия, които са слабо свързани.
- Теглото W_d на всеки клъстер е съобразено с удобството на съпровождане и намаляването на склонността за грешки.

Успешното прилагане на евристичен алгоритъм към проблема за клъстеризация на софтуер предполага:
- Правилно определяне на ресурсите и връзките в изходния код
- Подходяща формализация
- Подходяща дефиниция на целева функция и ограничителни условия. Типичните алгоритми, "базирани за търсене" извършват многократно изчисляване на целевата функция, следователно скоростта на изчисляването и е критично.
- Подходящ решение за различните критични елементи на алгоритъма: избор на <начално състояние>, на <съседно състояние> и др.

4.1. Определяне на ресурсите и връзките в изходния код

Определят се в зависимост от конкретизацията на поставената задача. В случая:

Ресурси: множеството от компоненти на ниско архитектурно ниво - модули, класове, файлове, пакети и др.;

Връзки: множеството от всички зависимости между компонентите - "наследява", "внася", "включва", "извиква", "е екземпляр на", и др.

4.2. Формализация

Цел на формализацията – независимост от програмния език, на който е написана системата, подлагана на клъстеризация. Представянето на код структурата на софтуерните системи най-често става чрез графи, като езиково независими структури. Алгоритми за клъстеризация се използват най-вече, когато тези графи станат огромни.

Дефинира се ориентиран граф G=(X,U), където:

X (множество от върхове) – представя множеството от ресурси (компоненти на ниско архитектурно ниво - модули, класове, файлове, пакети и др.);

$U \subseteq X \times X$ (множеството от ориентирани дъги) представя множеството от всички зависимости между ресурсите на кода, представени чрез X (зависимости между компонентите - "наследява", "внася", "включва", "извиква", "е екземпляр на", и др.).

Теглото - w_i, за $\forall x_i \in X$ изразява количествено характеристика на представения ресурс, например - броя на подпрограмите в модул x_i (брой на методите в обект x_i).

4.3. Характеристики на SOAD

- SOAD свежда проблема за клъстеризация до оптимизационния проблем за балансирано разрязване на графи [1], за който е характерно:
  - Много са възможните решения за изследване.
  - Намирането на оптималното решение е проблем от тип NP, което значи че най-добрите алгоритми изискват време, което е експоненциална функция на N=|X| и M, и че не съществува оптимален алгоритъм, чието време на изпълнение е полиномна функция на N=|X| и M.
  - Въвеждането на ограничения за баланс (в случая W_0), прави задачата още по-трудна (NP-hard).
  - Нужни са добри евристики, които "бързо" (тоест, за разумно време) намират един "приемлив", суб-оптимален резултат.

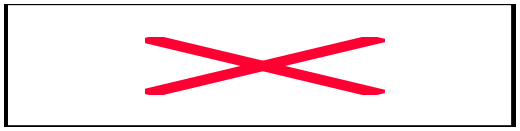
- Целева функция на SOAD.

Целевата функция е количествена мярка за качеството на всяко декомпозиционно решение.

Нека е намерено декомпозиционното решение $r \subset R$, получено при разрязване на графа G=(X,U) на M>1 подграфи, тоест:

$\exists r = \{X_1, X_2, \ldots, X_M\}$, $X_d \subset X$, $U_d \subset U$, $|X_d|=n_d$, d=1..M за което е в сила:

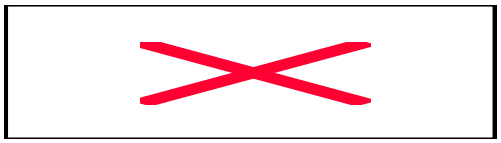
$(\cup X_d)= X$, d=1..M & $(\cap X_d)=\varnothing$ &



за $\forall X_d \subset X$, d=1..M е в сила ограничението (1):

…(1)

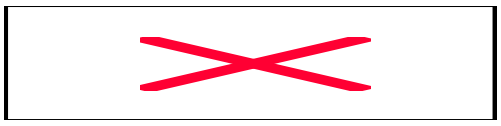
Нека $k_{df}=|U_{df}|$ е броят на външните връзки (на разсечените дъги) между подграфи $g_d \subset G$ и $g_f \in G$, където $d \in 1..M$, f=1..M, $d \neq f$.



Тогава общият брой "k" на външните връзки за системата се определя чрез израз (2):

…(2)

Тогава задачата за оптимално разрязване на граф G=(X,U) по указаните критерий и ограничително условие се състои в намирането на такова деление - $r_c=\{X_1, X_2 \ldots X_M\} \subset R$, M>1 за което се изпълнява описаното по-горе ограничение, така че функционалът (3)



…(3)

да достига своя минимум.

- SOAD – обобщен алгоритъм.

SOAD е разработен като алгоритъм за балансирано разрязване на

графи от смесен тип, с последователна и итерационна части. Последователната част – 3 балансирани по тегло деления $\{r_s, r_{ssh}, r_{long}\}$, базирани на известните от теория на графите алгоритми за обхождане на графи - случаен обход, обход в ширина, обход в дълбочина. Делението, което удовлетворява в най-голяма степен целевата функция, при спазване на ограничителното условие се избира за априорно:

$r_{apr} \in \{r_s, r_{ssh}, r_{long}\}$, при $k_{apr}=MIN\{k_s, k_{ssh}, k_{long}\}$.

Итерационната част на SOAD стартира с априорното решение - r_{apr}. Тази част е разработена като алчен градиентен алгоритъм. За целта, при всяка итерация се използват 4 подобряващи <текущото състояние> операции: "сливане на клъстери" (обединение на подграфи), "преместване на връх", "преместване на блок" или "размяна на единични върхове", тоест <съседно състояние> се получава от предното, при прилагане на коя да е горе изброените операции.

Стъпка 1. Вход на алгоритъма:
Структурен граф G=(X,U), N=|X|,
W – тегло на върховете.

Стъпка 2. Проверка за структурна коректност на графа.

Стъпка 3. Въвеждане на ограничителното условие - W_0.

Стъпка 4. Реализация на последователна част (последователен алгоритъм): $r_{apr} \in \{r_s, r_{ssh}, r_{long}\}$,
при $k_{apr}=MIN\{ k_s, k_{ssh}, k_{long}\}$.

Стъпка 5. Изпълнение на итерационна част
Стъпка 5.1.Нека $\mathbf{r_c}=\mathbf{r_{apr}}$; $\mathbf{k_c}=\mathbf{k_{apr}}$.
{ $\mathbf{r_c}$ - суб-оптимално решение}
Стъпка 5.2. Repeat
Намиране на най-добро <съседно състояние - r> за r_c, чрез прилагане върху всички двойки подходящо подредени съседни подграфи, на операции:

- "Сливане на клъстери".
- "Преместване на връх", "Преместване на блок"
- "Размяна на единични върхове".
- Ако ∃ такова <съседно деление - **r**>, тоест ако $k_r<k_c$ то
- $r_c=r$; Change=True; $k_c=k_r$;

Until not Change; {До – липса на промени върху структурата $\mathbf{r_c}$ (G^c)}
Стъпка 6. Извеждане на $\mathbf{r_c}$ - суб-оптимално решение за клъстеризация със стойност на целевата функция - $\mathbf{k_c}$, **dist** – дистанция между $\mathbf{r_{apr}}$ и $\mathbf{r_c}$.
Стъпка 7. Край.

- Основни операции и сложност на алгоритъма

Сложността на SOAD зависи основно от сложността на итерационната част. Последната от своя страна се определя от сложността на операция "размяна", която е в най-вътрешния цикъл. Операция "размяна на единични върхове" е разширение на водещата подобряваща евристика на Kernighan /Lin [1], това е алгоритъм за деление на ориентирани графи на повече от 2 части, с наличие на тегла на върховете, и собствена метрика за оценка на годността на върховете за операция размяна. Максимална сложност на операция "размяна" за двойка подграфи - $O((n_d+n_f)^3)$, следователно за всички двойки - $O(k_{apr} \times N^3)$, тъй като броят на инцидентните двойки подграфи на практика не може да надвишава броя на разсечените дъги. Следователно, сложността на SOAD е $O(k_{apr} \times N^3)$.

5. Заключение

SOAD е структурен, базиран на метрика алгоритъм за клъстеризация, реализиран като евристичен алгоритъм за балансирано разрязване на натоварени графи на произволно количество части – задача, която се обявява за слабо изследвана в теория на графите (класическите евристики са за

деление на графи с ненатоварени върхове, на 2 равни по тегло части) [1].

Основен недостатък на SOAD е силната зависимост на получения резултат от априорното декомпозиционно решение, който може да бъде преодолян за сметка на едно допълнително разширение, например увеличаване броя на декомпозиционните решения в последователната част.

Сравнението с алгоритмите за клъстеризация на софтуер SAHC и NANC [3,4] показва, че SOAD решава по-сложно формулирана задача и дава резултат - стабилен при всяко изпълнение и по-близък до оптималния:

- Стабилността на решението се дължи на: Базиране на итерационната част на SOAD върху априорно решение, което не случайно генерирано.
- За разлика от SAHC и NANC, разработеният алгоритъм е опит за решаване на задачата за деление на граф при натоварени върхове и ограничение – W_0.
- Използването на четири подобряващи операции върху двойка подграфи (спрямо една операция) предоставя условия за съществено подобряване на текущото решение в рамките на една итерация и едновременно с това – за по голяма близост до оптималното решение.
- Начинът на подредба на инцидентните подграфи и редът на изпълнение на операциите (последна е операция “размяна на единични върхове”) върху тях, изборът на високо “годни” върхове в операции размяна и преместване и “класическата” целева функция създават условия за редукция на аналитично намерената сложност на SOAD.

Основен недостатък на SOAD е силната зависимост на получавания резултат от априорното декомпозиционно решение, който може да бъде преодолян за сметка на едно допълнително разширение, например увеличаване броя на декомпозиционните решения в последователната част.

За експерименталното изследване на SOAD е създадена програма на Delphi 5 – прототип на инструментално средство за клъстеризация на софтуер [5]. Резултатите от експерименти върху голям брой крайни, ориентирани графи, чрез програмата доказват ефикасността и ефективността на разработения алгоритъм и маркират някои насоки за едно бъдещо развитие, например по отношение на определяне и обработка на “специални” върхове.

Използвана литература

[1]. Alan Pothen “Graph partitioning algorithms with applications to scientific computing”, Institute for Computer Applications in Science and Engineering (ICASE), NASA Lamgley Research Center, http://citeseer.nj.nec.com

[2]. John Clarke, Jose Javier Dolado “Reformulating Software Engineering as a Search Problem”, http://www.discbrunel.org.uk/seminal

[3]. Spiros Mancoridis and Brian MitchellComparing the Decompositions Produced by Software Clustering Algorithms using Similarity Measurements, IEEE Proceedings of the 2001 International Conference on Software Maintenance (ICSM'01)

[4]. Spiros Mancoridis, Brian Mitchell, C. Rorres, Y. Chen, and E. R. Gansner, Using Automatic Clustering to Produce High-Level System Organizations of Source Code, IEEE Proceedings of the 1998 International Workshop on Program Understanding (IWPC'98)

[5]. В.Т.Божикова, М.Н.Карова “Създаване, визуализация и операции на програмни структури”, 813-819, Proceedings of the Int’l Scientific Conference on Energy and Information Systems and Technologies, Vol.3., Bitola, June 7-8, 2001

ПОДОБРЕН АЛГОРИТЪМ ЗА МАРКИРАНЕ НА НЕПОДВИЖНИ ИЗОБРАЖЕНИЯ

Георги Б. Върбанов

Резюме: В доклада се предлага подобрен метод за маркиране на неподвижни цветни изображения с водни знаци. Реализираният метод е сляп алгоритъм-не изискващ наличието на оригинално изображение за откриването на наличието на воден знак. Алгоритъма е получен в резултат на взаимодействието на два отделни алгоритъма(сляп и не сляп). В резултат получен е много по-добър алгоритъм съчетаващ достойнствата и на двата алгоритъма. Този алгоритъм е значително по-устойчив на повечето деформации и JPEG компресия.

Improved vatermarked algorithm for still images

Georgi B. Varbanov

Abstract: In this paper we offer one improved method for watermarks still colors images. The proposed method is a blind algorithm-don't required in the presence of original picture for detection watermark. This is getting by interaction a blind with non-blin methods. The result in this get up an algorithm improved both merits. This is method is robustness in the most cases distortions and JPEG

1. **Увод**

С развитието на интернет и развитието на нови мултимедиини технологии направи въпросите за защита на информацията и авторските права все по-актуални. В настоящият момент все още съществува голямо разнообразие от методи за защита на информацията и авторските права. Няма разработен стандарт за тестване на създаваните алгоритми. Има няколко програми създадени от отдел-ни автори както и се работи по проект за създаването на стандарт в няколко европейски държави. Като основен инструмент се е наложила програмата StirMark създадена от [ ]. Най-честите атаки на които са подложени различ-ните методики са :

- Загуби направени от компресия
- Добавка на гаусов шум
- Медианно филтриране и размазване на контурите на атакуваните изображения,
- Пре-мащабиране на атакуваното изображение с последващо подрязване и група от атаки които целят да се премахне синхронизацията на водния знак за да не може да бъде извлечен.

Разнообразието на атаките, с които се въздейства с цел подправяне, подмяна или премахване на вложената защита е много голям. Много са малко или почти няма методи, които да покриват еднозначно повечето условия. В доклада се предлага метод разработен в резултат на влиянието на две методики с различни приложения и даващи различни резултати като в резултат на измененията се получават подобрени свойства. Разработени са няколко подобрени алгоритъма. Обект на настоящата статия е един от тях. За негова база са използувани non-Blind

адаптивния алгоритъм на Su, Wang и Kuo [6] изискващ за откриването на водния знак оригиналното изображение и този на алгоритъма, изследван от Tsekeridou и Pitas [2]

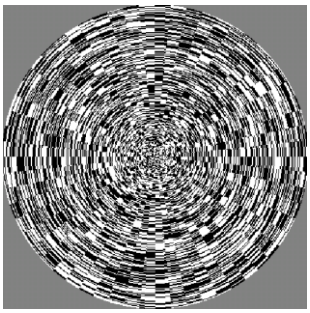
2. Подобрен алгоритъм на Wang, Su, Kuo, Tsekeriodu и Pitas

Първият метод, който е реализиран, е комбинация между алгори-тъма, изследван от Tsekeridou и Pitas [2] и алгоритъм, разработен от Su, Wang и Kuo [6]. От [2] е използвана идеята за генериране на себеподобен воден знак. Това става чрез генериране на воден знак, който реализира пространствено себеподобие в декартовата равнина. Това себеподобие ни осигурява изключително висока устойчивост срещу много видове атаки, като изкривявания, линейно и нелинейно филтриране, ротация, изрязване, скалиране, JPEG компресия. Този метод използва Wavelet преобразуване за да се получи висока надеждност и визуална нерушимост на изображението от направените промени. Извличането на водният знак става без наличието на оригиналното изображение, т.е. това е blind алгоритъм. Извличането става чрез статистическо изчисление на нормализирана зависимост на маркираното изображение от водният знак и сравнение с предварително избрана стойност, такава че да не допуска фалшива тревога за наличие на воден знак, или да не открива вграден воден знак. Основната част от настоящият алгоритъм е генерирането на водният знак. Той е така подбран, че самата му структура да осигурява изключителна защита срещу скалиране и изрязване на маркираното изображение. Водният знак представлява кръгово-симетричен сигнал. Този сигнал е реализиран от следната функция(1)[2]:

$$W_M(r,\theta) = \begin{cases} 0, & \text{if } r < r_{min} \text{ or } r > r_{max} \\ \pm\alpha, & \text{if } r_{min} < r < r_{max} \end{cases} \quad (1)$$

където

$r = \sqrt{n_1^2 + n_2^2}$, (n_1,n_2) представляват пространствените координати на пиксела, а $\theta = \arctan(\frac{n_2}{n_1})$ r_{min} и r_{max} са минималният и максималният радиус които описват големината на водният знак, α е цяло число, представляващо нивото на вграждане. За да бъде функцията $W_m(r,\theta)$ устойчива на компресия или филтриране, сигналът се разделя допълнително на s сектора, всеки с дъга от s/360 ^{o}. Всички пиксели в един сектор, имащи един и същ радиус се приравняват на – α или + α, в зависимост от псевдослучаен генератор на случайни числа, инициализиран със начална стойност k. Сигналът, описващ самоподобният воден знак има следният вид:



фиг.1

Това разделяне на сектори се прилага за да се противопостави на атаките от ротация. Откриването на водният знак при ротация на ъгли, помалки от s/360 ^{o} се извършва без да се завърта водният знак, а когато изображението е завъртяно на повече от s/360 ^{o} тогава за откриването на водният знак е необходимо да се завърти водният знак на ъгъл, кратен на s/360 ^{o}. Следващата стъпка е да се вгради себеподобие в отношение с декартовата равнина. Сигналът $W_m(n_1,n_2)$ се скалира (с използване на

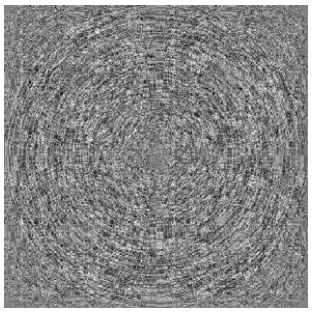
интерполация със съседни стойности), след което се измества спрямо центъра r=0 4 пъти. Това позволява да се открие видният знак при преоразмеряване на картината от 1/8 до 8 пъти. Така крайният воден знак се състои от 4 слоя :

$$W(n_1,n_2) = \sum_{f=0}^{3}\sum_{i=0}^{2^f-1}\sum_{j=0}^{2^f-1} W_M(2^f n_1 + i\lceil\frac{r_{max}}{2^f}\rceil, 2^f n_2 + j\lceil\frac{r_{max}}{2^f}\rceil)$$

(2)

Вижда се, че размерът на водният знак е $2r_{max}x2r_{max.}$ Параметрите, които са ни необходими за детекцията на водният знак са {k, α, r_{min}, r_{max}, s}.

Крайният резултат за воден знак изглежда така :



фиг. 2

Вграждането на водният знак се извършва със адитивен алгоритъм, разгледан подробно в [6]. В най-общи линии вмъкването на водният знак изглежда по следният начин :

фиг. 3

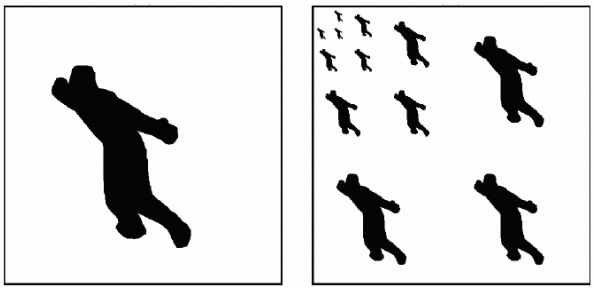
Изображението се декомпозира 2 пъти с използване на Haar Wavelet трансформацията. Така се получават общо 8 нива на декомпозиция от двете нива на Haar преобразуването, които се обозначават съответно с LL_1, LH_1, HL_1, HH_1, LL_2, LH_2, HL_2, HH_2 – съответно нискочестотни коефициенти от първото ниво, средночестотни коефициенти от първо ниво – два квадранта и високочестотни коефициенти, и съответните им коефициенти от второто ниво на декомпозиция. Тъй като най-много информация за картината се съхранява в нискочестотните изображения, за да можем да вкараме водният знак така, че да не е видим, коефициентите на водният знак се наслагват към нискочестотните и средночестотните коефициенти. Освен това се прави допълнителна маскировка, за да се изпълни адаптацията на изображението и да се замаскират вмъкнатите битове, така че да не са видими и да не могат да се открият чрез сканиране на честотите на изображението или по някакъв друг метод за анализ на цифрови сигнали. Както вече беше пояснено, първата стъпка към това е генерирането на водният знак, който се получава от налагането на функцията $W_M(x,y)$, която има стойности {-1, 1}, и има

средна значеща стойност 0. Тази функция представлява т. нар. Гаусов шум, който се налага към изображението и го прави практически неоткриваем, тъй като генерирането му става с помощта на псевдослучаен генератор на случайни числа. Допълнителното маскиране, което се прилага на към картината, е резултат от прилагането на алгоритъма ОПИ – Области, които Представляват Интерес. Това са тези области, които съдържат по-голямата част от информацията. Тези области се определят с помощта на принципа за MTWC – Multithreshold Wavelet coding, или многопрагово wavelet кодиране. Този принцип описва извличането от най-значещите коефициенти от даден вектор, който ги описва, като най-значещ означава такъв коефициент, който има най-голяма амплитуда. Общо казано тези коефициенти не се променят значително под въздействието на атаки от типа на компресия или геометрическа обработка. Ако тези коефициенти бъдат променени значително под някакво въздействие, то тогава изображението би било много по-различно от това, в което сме вградили водният знак.

Определението за най-значещи области има следният практически вид

фиг. 4

чрез селекция на най-значещите битове ние реално избираме само тази част от картината, която съдържа най-важната информация, която в горният пример е фигурата на сърфиста.



фиг. 5

След прилагане на Wavelet преобразуването, изображението, в което ще влагаме стойностите на водните знаци ще съдържа само стойностите на областта в която се намира сърфиста.

Алгоритъмът за определяне на коефициентите се изпълнява в следните стъпки :

Инициализация: установяване на праговата стойност T_s за всяка честота като половината от максималната абсолютна стойност на коефициентите от тази честота.

избиране честотна област, където максималният коефициент има стойност със стойност β x T_s, където β е коефициент, указващ силата на водният знак. За избраната честота се избират всички коефициенти, които имат стойност по-голяма от избраната честота T_s, и се маркират като значещи.

Водният знак се вмъква в избраните коефициенти от предната стъпка

Установява се нова стойност за $Ts^{new} = Ts/2$

Повтарят се стъпките от 2 до 4 докато се вмъкнат всички коефициенти от водният знак.

Формулата, която описва начина на вмъкване на водният знак има следният вид :

$$Cs'(x,y) = Cs(x,y) + y.\beta s.Ts.Wk \quad (3)$$

където C е избраният оригинален коефициент, C’ е маркираният коефициент, T_s е прагът на вмъкване за

текущата честота, W_k е стойността на водният знак {-α, α}, γ и β_s са коефициенти на квантоване. Стойността на γ се избира адаптивно, така, че да не се промени видимо картината. Този коефициент има смисъл да увеличава(намалява) маскирането на водният знак и да намалява (увеличава) сигурността на маркирането. Той се избира между (0,1). Параметърът β_s се използва да контролира избора на коефициентите в дадената честота – колкото по-голяма е стойността на β_s, толкова е по-голяма честотата, в която се вмъкват коефициентите. За извличане на водният знак се използва следната формула:

$$E^*_{s,k}(x,y) = C^*_{s,k}(x,y) - C_s(x,y), \quad (4$$

)където C_s* е коефициента, който сме извлекли от вероятно атакуваното изображение I*, и C_s е коефициента от оригиналното изображение I. Детекцията на водният знак се извършва чрез пресмятане на подобието между C* и C_s :

$$SIM(I^*, I) = \frac{\sum_{k=1}^{N_w} E^*_{s,k}(x,y) \cdot E_{s,k}(x,y)}{||E^*_{s,k}(x,y)||||E_{s,k}(x,y)||} \quad (5$$

)

където Nw е общият брой на символите от водният знак, $E_{s,k}(x,y)$ е оригиналният воден знак и $E^*_{s,k}(x,y)$ е водният знак, извлечен от коефициентите на атакуваното изображение. Важно е да се отбележи, че се използва скаларно произведение на два вектора, нормализирано с техните нормални стойности като мярка за подобие, което измерва косинусовата функция на ъгъла между двата вектора. Всяко подобие, по-малко от 0 се третира като нула, тъй като тогава тези два вектора са с противоположни посоки. Максималната стойност на това подобие е 1, когато имаме неатакувано изображение и водният знак е извлечен изцяло.

Въпреки че водните знаци, генерирани от псевдо-случайните последователности не могат да дадат подобра корелация от случайните числа с равномерно разпространение, енергията на различните водни знаци може да се направи подобна, така че да се разпростре в някакви граници. Тъй че, водни знаци, генерирани с различни инициализиращи стойности на псевдослучайният генератор могат да имат една и съща степен на устойчивост срещу атаки.

Описаното дотук описва т. нар. алгоритъм non-blind водно маркиране, което изисква наличието на оригиналното изображение, за извличане на водният знак. За да се преобразува в blind, се налагат някой трансформации на алгоритъма, за да може да се прави детекция, без наличие на оригиналното изображение. Най-трудният проблем при подобни схеми е да се определят коефициентите, в които е вмъкнат водният знак, както и да се отделят коефициентите на самият воден знак. Основната идея на “blind” алгоритмите е да се изрежат получените коефициенти на базата на генерирани “псевдо-оригинални” стойности, и тогава водният знак да се наложи върху тези коефициенти за да се получи т. нар. “защитено изображение”.

Нека $C_s(x,y)$ е избраният коефициент от честота s с текуща прагова стойност Ts, така че $Ts \le C(x,y) < 2 \times Ts$, тогава защитената стойност на C_s ще бъде :

$$C'_{s,k}(x,y) = sign \times \Delta_p(C_s(x,y)) + \alpha\beta_s T_s W(x,y)$$

(6)

където W(x,y) е стойността на водният знак, *sign* е знакът на стойността на $C_s(x,y)$, и операторът Δ е дефиниран като:

$$\Delta_p(C_s(x,y)) = (1 + 2p\alpha)T_s,$$

(7)

където p е цяло число, между 1 и $(2\alpha)^{-1}$. Тогава разстоянието $DIS_{s,p}(x,y)$ между $\Delta(C_s(x,y))$ и $C_s(x,y)$ се определя от следната формула(9):

$$DIS_{s,p}(x,y) = |\Delta_p(C_s(x,y)) - |C_s(x,y)||$$

(8)

Така че вече можем да определим стойността на p чрез :

$$p = \arg\min_{p'} DIS_{s,p'}(x,y)$$

(9)

След като сме определили p, имаме :

$$DIS_{s,p}(x,y) \le \alpha \times T_s$$

(10)

Детекцията на водният знак се изчислява по дадената по-горе формула, където $E_{s,k}(x,y)$ се замества с:

$$E^*_{s,k}(x,y) = C^*_{s,k}(x,y) - sign \cdot x \cdot \Delta_n C^*_{s,k}(x,y)$$ (11)

където Δ_p е равно на $(1+2p\alpha)T_s^*$, и Ts се изчислява от C*. Тъй като Ts* се изчислява на базата на най-големият коефициент в дадена честота s, и в този коефициент не се вмъква никаква допълнителна информация, може да се счита, че той ще бъде максимално близък до оригиналният. Ако се вземе за даден коефициента β <= 1.0 и ако стойността на коефициента $C'_{s,k}(x,y)$ е била променена с по-малко от αT_s, тогава водният знак в $C_s(x,y)$ може да бъде извлечен изцяло. По големи стойности на α могат да осигурят по-голяма устойчивост на водният знак, но също така и по-голяма стойност на шума, генериран от водният знак PSNR.

6. Заключение

Предлаганият метод може да се използува експерименти по разработването на ефективни алгоритми за поставяне на водни знаци. В резултат така променения алгоритъм в значителна степен е по устойчив на геометрични изкривявания от повечето познати блинд алгоритми, подрязване завъртане и не изисква за извличане на водния знак необходимост от оригиналното изображение.

Литература

[1] V. Solachidis and I. Pitas, “Circularly symmetric watermark embedding in 2-d dft domain”, in *Proc. of ICASPP’99*,Phoenix, Arizona, USA, 15-19 March 1999.

[2] S. Tsekeridou and I. Pitas “Wavelet-based self-similar watermarking for still images”

[3] C.-T. Hsu and J.-L. Wu, “Multiresolution watermarking for digital images”, IEEE Trans. on Circuits and Systems II, vol. 45, no. 8, pp. 1097–1101, August 1998.

[4] J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, “Secure spread spectrum watermarking for multimedia,” IEEE Trans. Image Processing,

[5] Liang J., Xu P and Tran T.D. – A universal robust low frequency watermarking scheme

[6] Po-Chyi Su, Houng-Jyh Mike Wang and C.-C. Jay Kuo, Digital Image Watermarking in Regions of Interest

[7] Liehua Xie, Gonza lo R. Arce – Blind wavelet based digital signature for image authentication, Proceedings of the 9th European Signal Processing Conference, EUSIPCO ‘98

[8] Liehua Xie, Gonzalo R. Arce – Joint wavelet compression and authentication watermarking - Proceedings of IEEE International Conference on Image Processing ICIP ‘98

[9] Petitcolas F. Anderson R. J., Kuhn M., Attacks on Copyright Marking systems, Second workshop on information hiding, in vol. 1525 of Lecture, 1998

[9] Voloshynovskiy Sv., Herrigel Al., and Pun Th., Blur/Deblur attack against document protection systems based on digital watermarking

[10] Sviatoslav Voloshynovskiy, Shelby Pereira, Alexander Herrigel, Nazanin Baumgartner, Thierry Pun - Generalized watermarking attack based on watermark estimation and perceptual remodulation, University of Geneva, Department of Computer Science,

[12] Kundur D., Hatzinakos D., A robust digital image watermarking method using

wavelet-based fusion. Proceedings of the IEEE ICIP, 1997
[13] Koch E., Zhao J., Towards robust and hidden image copyright labeling, Proceedings of ICIP of the IEEE International Workshop on Nonlinear Signal and Image Processing, 1995
[14] Рабинер, Б.Гоулд, Теория и применение цифровой обработки сигналов. М. “Мир” 1978г.

Комуникационна среда на система за разпределена симулация

Х.Вълчанов, Н.Рускова, Т.Русков

Резюме: Разпределеното дискретно моделиране съкращава времето за изследване и анализ на сложни дискретни системи чрез използване на изчислителната мощ на множество процесори. Статията представя структурата и организацията на подсистема за комуникация като част от система разпределено моделиране, функционираща под управлението на ОС Linux в средата на мрежа от работни станции. Представен е вариант на клъстерна организация на логически процеси, като е представен алгоритъм за комуникация между работните станции.

Comunication Subsystem for Distributed Simulation Environment

H.Valchanov, N.Ruskova, T.Ruskov

Abstract: Parallel discrete event simulation (PDES) is a basic approach for evaluation of the complex systems. A PDES attempts to speed up the execution of a simulation by distributing the simulation's workload between multiple processors. A network of workstations is a widely available platform for PDES. This paper explores an implementation's details of a communication subsystem for PDES. The techniques for reducing of interprocessor communication overload are presented.

1. Увод

Основен подход за комплексно изследване и оценка на поведението на сложни системи е имитационното моделиране (ИМ), известно още като симулация. За постигането на точни резултати е необходимо разработване на обемни и комплексни модели, което от своя страна води до драматично нарастване на времето за тяхната интерпретация. Ускоряване на процеса на моделиране може да се постигне чрез неговото разпаралелване (разпределяне) върху множество процесори в рамките на мощни мултипроцесорни системи. Поради тази причина разпределеното имитационно моделиране (РИМ) се развива в ограничен брой научно-изследователски лаборатории, разполагащи с мощни компютри, изградени на базата на десетки и даже стотици паралелно работещи процесори.

При РИМ моделът се декомпозира на логически процеси (ЛП), всеки от които симулира един или няколко компоненти на моделираната система. ЛП се изпълняват върху различни процесори. Взаимодействието между компонентите на системата се отразява чрез събития, които се симулират посредством предаване на съобщения между логическите процеси. Всяко съобщение носи информация за времето на настъпва-не на съответното събитие (временен маркер) [1].

Разпространението на локалните компютърни мрежи (ЛКМ) дава нови възможности за организация на големи изчислителни ресурси за ускоряване на симулационния процес, както и за популяризирането му. Типично мрежите са изградени от работни станции на

базата на персонални компютри, свързани чрез Ethernet протокол и се разглеждат като системи с разпределена памет. ЛКМ притежават ниска себестойност и възможност на лесна преконфигурация за разлика от традиционно използваните за целите на РИМ мултипроцесорни системи с обща памет. Същевременно ЛКМ използват масово разпространено апаратно и програмно осигуряване, което ги прави достъпни за широк кръг потребители. Разпределените системи, базирани на ЛКМ, позволяват ефективно използване на наличните ненатоварени процесори и увеличаване на компютрите в мрежата, създавайки условия за изпълнение на симулационни модели с достатъчно висока степен на паралелизъм [2].

2. Комуникация между ЛП

При междупроцесната комуникация в системите с разпределена памет, каквито са ЛКМ, съобщенията за събития се предават между процесорите в мрежата с помощта на подходящи мрежови протоколи. Използваните протоколи за синхронизация изискват множество служебни съобщения, които натоварват допълнително мрежовата комуникация. Ограничената пропуска-телна способност, закъсненията за транспортиране на съобщенията и необходимостта от надеждни протоколи за мрежова комуникация между работните станции определят и по-ниската ефективност на РИМ в ЛКМ по отношение на системите с обща памет.

За постигане на висока производителност, разпределените системи за моделиране са базирани на библиотеки за предаване на съобщения. Един от най-често използваните съвременни комуникационни стандарти е стандартът MPI [3]. Стандартът MPI дефинира програмен интерфейс, базиран на обмен на съобщения. В модела на MPI приложната програма се разглежда като комплекс от комуникиращи си паралелни процеси, обособени в отделни групи. При комуникацията се използва механизъм за буфериране на съобщенията, който е прозрачен от гледна точка на потребителя. На мрежово ниво в повечето реализации обменът е базиран на протокола TCP/IP. MPI предоставя удобни от гледна точка на програмиста средства за обмен на съобщения, но специфичният характер на синхронизацията и използването на блокиращи примитиви за обмен не съответстват на асинхронния характер на комуникацията в РИМ и може да предизвикат ситуации на взаимна блокировка между процесите.

Настоящата статия дискутира особеностите на изграждането на ефективна комуникационна подсистема като част от разпределена система за имитационно моделиране, функционираща върху локална мрежа от персонални компютри i586 под управлението на ОС Linux и използваща транспортен протокол SCTP.

3. Комуникационна среда

Комуникационната подсистема (КП) (фиг.1) на системата за разпределено симулиране [4] реализира комуникационния интерфейс в разпределената изчислителна среда. Комуникацията между работните станции е реализирана на базата на протокола **SCTP** (Stream Control Transmission Protocol), намиращ се на транспортното ниво на TCP/IP протоколния стек [5].

Фиг.1 Структура на система за РИМ

С цел повишаване на ефективността на моделирането е въведена клъстерна организация на логическите процеси. Определен брой ЛП се групират в рамките на един процес в ОС, изпълняващ се на дадена работна станция. Този процес (клъстер) осъществява планирането на своите логически процеси и комуникацията с останалите клъстери на същата работна станция и в мрежата. Всеки клъстер включва специално комуникационно ядро (КЯ).

Комуникационните ядра комуникират помежду си чрез SCTP съобщения, които според своята фунционалност се разделят на две основни групи:

- даннови съобщения – съобщения, използвани за целите на симулацията. Такива съобщения са изпращат от даден ЛП към входящата опашка на друг ЛП;
- контролни съобщения - тези съобщения остават скрити за приложния код. Това са съобщения, които ядрата пращат помежду си, за да синхронизират работата си и да контролират процеса на симулация.

4. Архитектура на комуникационната среда

Възможни са две основни архитектури за реализация на взаимодействието между работните станции:

- разпределена архитектура - всички работни станции са равностойни и във всеки един момент от време дадена работна станция може да изпрати съобщение до всяка друга работна станция;
- централизирана архитектура - тази архитектура предполага съществуване на координираща работна станция. Всички съобщения се изпращат първо към тази станция, която интерпретира или препраща към станциите получатели.

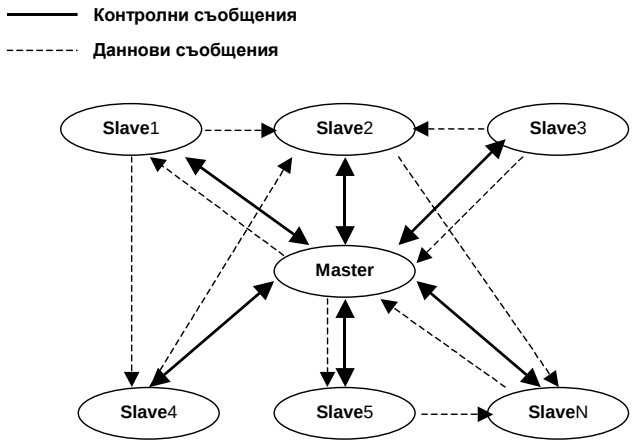
Всяка една от тези архитектури има своите предимства и недостатъци. Първата архитектура е сравнително лесна за реализация и позволява принципно бърз обмен на данни между комуникационните подсистеми на работните станции. Динамичното създаване на нови ЛП, динамичното балансиране на натоварването, възможността да се изпраща съобщение до ЛП, изпълняващ се в произволен клъстер, води до необходимостта всяка една работна станция да съхранява пълна информация за състоянието на останалите работни станции. Това от своя страна значително увеличава мрежовия трафик, увеличава времето за обработка на постъпилите от мрежата съобщения и води до забавяне на работата на системата, поради факта, че всяка работна станция при промяна на своето състояние трябва веднага да изпрати съобщение до другите.

При втората архитектура КЯ на координиращата работна станция работи по алгоритъм, различаващ се от алгоритъма на останалите станции. Единствено координиращата станция съдържа пълната информация за всички останали и играе ролята на координатор между тях. Това значително намалява трафика по мрежата, но същевременно увеличава времето, за което КЯ на две станции си обменят информация, тъй като съответните съобщения трябва да преминат през координатора.

Разработената комуникационна подсистема използва смесена архитектура, като се опитва да ползва предимствата и на двата алгоритъма и да изключи някои от недостатъците им. Предимствата на разпределения алгоритъм по отношение на данновите съобщения са безспорни. Данновите съобщения възникват асинхронно, не изискват задължителен отговор и нямат пряко отношение към работата на комуникационната система. Те се пращат директно между работните станции. При контролните съобщения нещата стоят по-различно. За генерирането на правилно контролно съобщение станцията-източник трябва да има определена информация за текущото състояние на цялата система. Например, при генерирането на съобщение за създаване на нов ЛП е необходимо работната станция да познава всички работни станции, участващи в симулацията. Освен това, за реализиране на балансирано динамичното натоварване, станцията трябва да има информация за моментната натовареност на останалите станции, за да определи най-подходящата за създаване на новия ЛП. В този случай предимствата сочат към централизирания алгоритъм.

За да се съчетаят предимствата на двете архитектури, при реализацията на комуникационна подсистема е избран принципът "главен-подчинен" (*master-slave*). На всички работни станции, участващи в симулацията, се зарежда едно и също комуникационно ядро, като една от работните станции се избира за главна (*master*). Тя изпълнява всички функции, които изпълняват и другите ядра, заедно с функциите, нужни за координация на контролните съобщения и манипулиране с информацията за състоянието на системата.

На фиг.2 данновите съобщения между ЛП се пращат директно към клъстера, в който е получаващият логически процес. Всяка промяна на своето състояние даден клъстер съгласува чрез контролни съобщения с ядрото *master*. По този начин, във всеки един момент на симулацията, *master* има пълната информация за състоянието на системата.



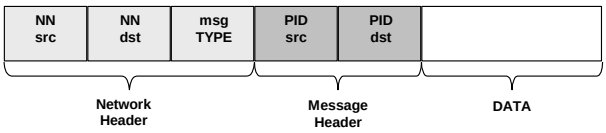
Фиг.2 Взаимодействие на комуникационните ядра

Той може да вземе адекватно решение за създаването на нов ЛП като същевременно има готовност за обслужване на заявки от другите клъстери за информация, отнасяща се до системата. Всички останали клъстери не е необходимо да имат информация за процесите, изпълня-ващи се извън тях. При нужда за получаване на информация за дадена част от системата, съответният клъстер изпраща заявка във вид на контролно съобщение към *master*, който му отговаря с ново контролно съобщение-отговор.

4. Адресиране на ЛП в системата

За да се постигне нужното ниво на абстракция от мрежовия слой, на всеки клъстер, участващ в процеса на симулация, се присвоява уникален номер, наречен Node Number (*NN*). *NN* е 4 байтово число (фиг.3). Адресация на даден клъстер в рамките на системата се извършва чрез неговия

NN, който еднозначно го идентифицира. В мрежовия модул на комуникационното ядро се прави съответствие между *NN* и мрежовия адрес на Linux машината, в която той се изпълнява.



Фиг.3 Формат на обменяните съобщения

Адресацията на логическите процеси става чрез идентификатора на логическия процес (PID).

Всяко съобщение, независимо от това какъв тип е, притежава мрежова заглавна част - Network header, която съдържа номерата на изращащия и приемащия клъстер, както и типа на съобщението – за данни или контролно.

5. Заключение

Разработената комуникационна подсистема е предназначена да организира ресурсите на клъстер от Linux машини, свързани в Ethernet LAN, за съгласувано функциониране в единна система за целите на разпределеното имитационно моделиране.

Системата е разработена на език C, функционира под управление на ОС Linux и притежава следните особености:

- използваната нишкова технология позволява допълнително увеличаване на ефективността при симулация за сметка на паралелно изпълнение на отделни части на системата;
- възможност за асинхронно изпращане и получаване на съобщенията за настъпили събития;
- използването на динамични модули дава възможност за преносимост на други мрежови платформи, както и добавяне на други протоколи за комуникация.

Предложената система може лесно да бъде модифицирана и за други разпределени приложеия. Предстои нейната интеграция във Web базиран програмен комплекс за разпределена симулация.

6. Литература

1. Fujimoto, R. Parallel Discrete Event Simulation. CACM, 1990, No.10, 30-53.
2. Hara, V., Harju, J., Ikonen, J. Simulation of Cellular Mobile Networks in a Distributed Workstation Environment, Annual Review of Communications, 1997, 913-917.
3. Message Passing Interface Forum. MPI: a message-passing interface standard, 1995.
4. Ruskova N., Walchanov H. Run-time system for a distributed simulation environment over network of workstations. Proc. of the CompSysTech'2000, Sofia, 2000, I.8-1 - I.8-6.
5. R. Stewart, Ch. Metz, SCTP. New Transport Protocol for TCP/IP, IEEE Internet Computing, November – December 2001.

Оценка на качеството на Web-базираните учебни курсове

Бойка Ж. Градинарова

Резюме: В доклада се предлага метод за оценка на качеството на Web-базирани курсове за обучение. Методът се основава на количествено измерване на качествените параметри отнасящи се до три аспекта на курса: продуктите създадени от участниците, съдържанието на курса и осъществената интерактивност. Оценяването се базира на изследване на обменените съобщения и продуктите създадени по време на учебния процес.Методиката обхваща набор от качествени параметри характерни за он-лайн обучението, таксономия на съобщенията базирана на тяхната комуникативна функция, както и набор от концептуални и технологични инструменти, които биха могли да се приложат. Разгледан е пример от експерименталното приложение на метода.

Evaluating the quality of Web-based courses

Boyka Z. Gradinarova

Abstract: A methodology for evaluating online collaborative learning processes is proposed. The underlying hypothesis is that three elements affect the quality of online courses: learning quality, content quality, and interaction quality. Evaluation derives from examination of all the messages and products that the online learning process yields. The methodology comprises a set of parameters that indicate the quality of online courses, a taxonomy of messages based on their communicative function, several modalities of approach, and a set of conceptual and technological tools that can be adopted. The paper presents a case study wherein the methodology is applied to an online course dealing with environmental education.

1. **Увод**

Обичайното електронно обучение освобождава обучаемите от нуждата да присъстват на определено място по определено време, затова то рядко предлага възможности за съвместно обучение. Поради тази причина стимулирането на интерактивността и сътрудничеството между участниците при този вид учебен процес са от съществено значение. Нещо повече, базираната на текст комуникация изразяваща се в интензивен обмен на съобщения, позволява целия процес на сътрудничество да бъде запазен и възстановяван. Това дава възможност за преглед на избраните методи и оценка на курса като такъв.

Предлаганият метод за оценяване процеса на електронно обучение се базира на отчитането на широк спектър от характеристики, касаещи качеството му. Целта е да се определи система за управление на качеството позволяваща не само да се оцени постигнатото от обучаемите и добитите от тях знания, но и да се определи дали курса действително е задоволил нуждите предизвикали създаването му. Изхождайки от предпоставката, че качеството на електронното обучение зависи от три елемента: качество на добитите знания; качество на съдържанието в зависимост от обмена на знания и дискусиите в които обучаемите се включват; и качество на интерактивността имайки предвид качеството на процеса на общуване по време на обучението. Така оценката се извлича от изпитните резултати, от горе опоменатите три елемента и от всички съобщения и продукти създадени по време на курса. Тя обхваща определен брой параметри

на качеството на учебния курс, класификация на съобщенията в зависимост от комуникативните им функции и някои теоретични и технологични инструменти, които могат да се приложат.

2. Комуникационна платформа

Комуникационната платформа осигуряваща средата за съвместно обучение трябва да позволява интерактивност за отдалечените в пространството потребители, това може да бъде чрез използването на електронна поща или познатите конферентни системи. При тях се ползва асинхронна комуникация, понякога включваща синхронни функции като чатинг и видеоконфериране.

Средата за компютърно конфериране се отличава с контролиран достъп. Тя осигурява следните интерактивни режими:

- Обмен на информация, споделяне на знания, групово проектиране и разработване на продукти;
- Изработване на общи решения и преговори;
- Инициативи по сближаване
- Достъп до външни ресурси: публични Интернет сайтове;
- Достъп до мултимедиен учебен материал

3. Структура на модула

При он-лайн обучението съвместната работа на групата изисква грижливо планиране, координиране във времето и синхронизиране на усилията на преподавателите, които играят основна роля по отношение на стимулирането, контролирането и регулирането на учебния процес. Учебните модули могат да се разделят на пет основни фази:

1. *Стимулиращи съобщения на преподавателя*
2. *Вътрешно групови дейности*
3. *Групови съобщения отговори*
4. *Анализ и покана за дискусия*
1. *Дискусия между различни групи*

4. Критерии за оценяване

Това което трябва да се оцени е качеството на он-лайн курсовете. Поради това най-напред следва да се дефинира какво разбираме под качество на учебния процес изобщо и на Web-базираното обучение в частност. За целта трябва да дадем една най-обща дефиниция за обучението: “ще приемем обучението като изменение в индивидуалното познавателно, психомоторно и емоционално състояние в резултат на процес предназначен да предизвика това изменение” (1). В същност този процес може да предизвика и непредвидени промени. Поради това оценката на качеството на обучение се извлича на базата на обща оценка на всички тези възникнали изменения съобразно набор от определени критерии.

Тези критерии могат да се различават в зависимост от приложената учебна теория. Така например за бихевиориста колкото повече резултатите се доближават до предварително набелязаните учебни цели толкова по успешно е обучението. От друга страна за конструктивиста по доброто обучение се състои в доближаване на автономното възпроизвеждане на знания от студента до резултати които определена общност счита за правилни; от тази гледна точка учебния процес трябва да генерира високо ниво мета когнитивни структури обогатяващи базата от умения на студента.

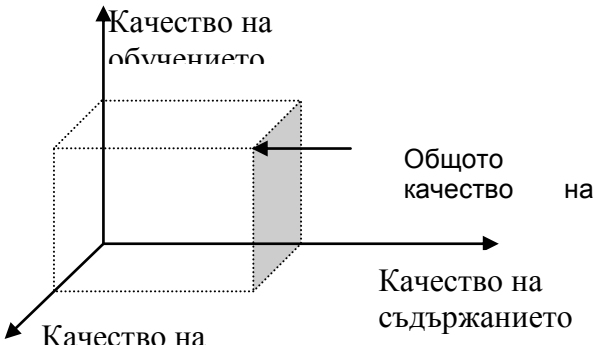
Тези примери дават възможност да дефинираме обучението най-общо като изменение в нивото на студента, като оставим дефинирането на качествените критерии на съответна учебна теория.

Он-лайн обучението е базирано основно на съвместни учебни стратегии. Като широка дефиниция на понятието "съвместна учебна дейност" може да се приеме индивидуалното придобиване на знания, умения или поведение в резултат от групово въздействие или накратко индивидуално обучение в резултат на групов процес вж.(2) и (3).

При електронното обучение взаимодействието най-често е насочено към определена област на знание. Качеството на това взаимодействие зависи от два основни елемента; качество на диалога развит в рамките на групата и качество на съдържанието на този диалог. При все, че качеството не е абсолютна величина, все пак то може да се прецени съобразно тълкуването от страна на оценяващия на контекста основаващ се на заложените цели на обучението.

На тази основа представяната методика сравнява качеството на он-лайн курсовете по три показателя, които могат да се приемат за независими компоненти в триизмерно качествено пространство (вж. Фиг.1):

1. Знания, постигнати от отделните участници
2. Взаимодействие, което ще бъде оценено във връзка с контекста в който се осъществява и на база на предварително набелязани цели;
3. Съдържание на курса: съдържанието на он-лайн курса, отчасти предоставено на участниците в него и отчасти доразвито от тях, трябва също да бъде оценено в зависимост от съдържанието на диалога между участниците



Фиг.1 Цялостно качество на учебния курс

3.6. Способ на оценяване

Повечето от методите за оценяване попадат в една от следните две основни категории:

1. Оценка на база на продукта.

Учебния процес създава обекти (решения на проблеми, продукти и т.н.) които биха могли да се оценят по вътрешни критерии (съответствие, естетика и т.н.) или външни критерии (точност, отговаряне на дадени условия и т.н.);

2. Оценка на база на представянето

Он-лайн курсовете винаги създават най-малко един "обект" например съобщенията създадени от членовете на учебната група. Освен това дейността на участниците често създава продукти (доклади, проекти, обобщения и др.) Съответно изглежда логично да приемем, че оценяването следва да се основава на тези "обекти" (съобщения или продукти) и да отчита както качеството на заученото (извлечено от индивидуалните продукти), така и качеството на процесите на взаимодействие осъществени в рамките на групата, което от своя страна е свързано както с качеството на съдържанието, така и на самото комуникиране.

Основната идея на която се основава представяната методика е да се степенуват съобщенията и продуктите съобразно предварително набелязани качествени параметри. Он-лайн

курсовете се състоят от определен брой конференции (сесии), съставени от потоци от съобщения, които от своя страна са съставени от индивидуални съобщения. Съответно качеството на курса като цяло се определя чрез сума от качествените оценки на конференциите. Качеството на конференцията се изчислява на база качеството на потоците, което от своя страна сумира качеството на отделните съобщения.

7. Класификация на съобщенията

Друг важен елемент на който се крепи методиката на оценяване е класифицирането на всяко съобщение съобразно предварително определена таксономия. Оценявайки качеството на съобщението ние не можем да не вземем предвид специфичното му място в комуникационния процес. Във връзка с това таксономията помага на оценяващия в класификацията на съобщенията във функционални категории. Възможната класификация на съобщенията изпращани като част от дистанционното взаимодействие се базира на пет основни категории (стимулиране, отговор, дискусия, координация и социално общуване), които на свой ред се разделят на под-категории. Все пак трябва да се отбележи, че тези категории не са взаимно изключващи се, но реално допълващи се. За да се избегне загубата на ценна информация, особено когато имаме работа с дълги и засягащи няколко теми съобщения може да се даде възможност за включване на съобщението в две (дори и в три) категории.

8. Инструментални и оперативни аспекти

В тази част се описват средствата (концептуални и технологически) и фазите, които обхваща методиката за оценяване. Този модел определя структурата на базата данни използвани при оценката на он-лайн курса.

Главните елементи на базата данни са:

- Членове на групата – които могат да бъдат индивидуални студенти, група или преподавател.
- Конференция, която определя отделния модул на курса.
- Съобщение, което обхваща определен брой атрибути:
  - функция, която изпълнява (убеждаваща, информативна и т.н.)
  - темата на съдържанието (КБК система, теми свързани с курса и т.н.)
  - нишката към която принадлежи съобщението; трябва да се отбележи, че в някои случаи тя може да съдържа само едно съобщение, когато същото не е получило отговор или коментар;
  - типа - това е най-широката категория базирана предимно на съдържанието;
  - качествени параметри – това са параметрите, които описахме по-горе.
- Документи.

Тази категория се различава от предходната по това, че документите съдържат материал, който не е непременно плод на комуникацията в курса и чиято цел надхвърля този контекст. Примери за такива документи са студентски продукти, референтни материали изготвени от студентите в дигитална форма, файлове във формат несъвместим с този на съответното съобщение (графики, електронни таблици, база данни и т.н.)

Документите често се прилагат към дадено съобщение и се разпределят в рамките на групата, в който случай основното отношение е очевидно. В много случаи обаче се предпочита документа да бъде отделен от съобщенията свързани с него и тогава отношението трябва да се изрази по друг начин.

Съставянето на базата данни се извършва полуавтоматично. Обективните данни като подател, получател и други свързани с всяко съобщение се влагат от КБК сървъра. От друга страна субективните данни като тип, функция и качество на съобщението се определят от оценяващия съобразно съдържанието на дискусията и се включват в базата данни чрез съответна форма. Количествените измервателни стойности се извличат от средно-претеглените, като всеки елемент помагащ за определяне на средната величина се измерва точно. Определяйки теглото на различните параметри оценителят може да хармонизира процеса на оценяване в съответствие с всяка от целите на конференцията. Например на параметъра спазване на системата от правила може да бъде придадено по-голямо значение при конференциите по зададена тема и по-малко значение при конференциите за събиране на референтни (справочни) материали.

След като дискриптивната информация за всяко съобщение бъде съхранена в базата данни заедно с данните отнасящи се до други величини, става възможно да се извлече и екстраполира информацията касаеща качеството на проведения учебен процес с помощта на КБК системата.

Например цялостното качество на интерактивността за курса се извлича от общата сума на качествените нива на интерактивност на отделните конференции. Факторите определящи измерването включват нивото на участие (броя съобщения изпратени от участниците), нивото на дискусията (брой на нишките) и преобладаващ тип на съобщенията (предложения, искания за помощ и т.н.)

Качеството на интерактивността на конференцията от своя страна зависи от качеството на нишките, но също и от фактори като тяхната продължителност, времетраенето им и преобладаващия тип на взаимодействие (студент-студент, студент-група, студент -преподавател, преподавател-група). По същия начин качеството на потока се извлича чрез сумиране на качественото ниво на всяко съдържащо се в него съобщение. Последният елемент касаещ качество-то на цялостната интерактивност на курса е качеството на съобщението, което се извежда от стойностите зададени на всеки от качествените параметри.

Качеството на интерактивността е само един от трите компонента определящи качеството на курса. Качеството на обучение (отнесено към продукцията на студента) се изразява чрез два качествени параметъра прилагани към всяка конференция; формално съответствие и съответствие на съдържанието.

Очевидно ползата от такава система не се ограничава само с изчисляването на определена числена стойност отразяваща цялостното ниво на частични качествени оценки като тези на две тематично базирани конференции или на една и съща конференция при две различни провеждания на един и същ курс.

2. Заключение

Експерименталното проведените тестове подчертават ефективността на подхода, потвърдена от сравнението с резултатите получени, чрез ползването на традиционни методи за оценяване.
В бъдеще системата за комуникиране би могла да бъде усъвършенствувана така, че студентите и оценителите да могат да определят смисъла на съобщението от момента на неговото създаване. Изпращащите съобщения ще имат на разположение меню, от което да могат да изберат комуникационната функция на съобщението си преди да го изпратят (напр. “ запитване”, “ коментар” , “предложение за дискусия” и т.н.). Това ще улесни комуникирането

между участниците и ще направи полека задачата на оценителите. Първата стъпка в това отношение е определянето на изчерпателна и отворена таксономия на съобщенията обхващаща комуникационните функции прилагащи се при он-лайн курсовете

Литература

1. Midoro, V. (1999) Modelling On-Line education. *Proceedings of Comned 99 IFIP Open Conference*, Aulanko, Finland, June 13 - 18.

2. Kaye, A. R. (1991) Learning together apart. In Kaye A. R. *Proceedings of the NATO Advanced Research Workshop on Collaborative Learning and Computer Conferencing, Series F: Computer and System Sciences '90*. Berlin:Springer-Verlag.

3. Briano, R., Manca, S., Midoro, V., Persico, D.&Sarti, L.(1999) On-Line Teacher Training in Environmental Education across Europe. In *Proceedings of the Telecommunications for Education and Training 1999 - TET99 Conferences.* Gjшvik, Norway, June 8 - 11.

Генериране на текстурни изображения посредством модулация на покрития на двумерна повърхност

Огнян И. Железов

Резюме: В тази статия се предлага алгоритъм за генериране на текстурно изображение посредством покриване на двумерна повърхност с многоъгълници от зададен вид с модулирани с избрана функция параметри. Модулацията на паркетиращите равнината многоъгълници дава като резултат различни по вид текстури, състоящи се от двумерни области, ограничени от контурни криви, получени чрез филтрация на паркетиращите многоъгълници. .

Generating Textures Algorithm using causal modulation of 2-D surface covering

Ognyan I. Zhelezov

Abstract: In this paper is proposed an algorithm for generating synthetic textures baset on modulation of poligons, that covers the two-dimensional surface. The modulation gives as a result a different textures, composed by two-dimensioned regions. Using different modulation finctions it is possible to create different pattern images. Additional parameters are the color of lines and fill color of rectangular element. Smoothing of poligon contours using LF filter gives more realistic objects containing textures.

1. Въведение

Текстурите се използуват в компютърната графика при синтезиране на обекти и компютърни сцени, за получаване на различни по вид повърхностни детайли. Дигитализирана текстура може да бъде получена от много източници, например от цифрва фотография, но полученото текстурно парче няма да има желаните размери или форма. Ако при покриване на по-голяма повърхност се използува просто повторение на текстурата, това води до неприемливи ефекти – забележими преходи, забележими повторения и т.н. Един от използуваните методи за получаване на текстурни покрития с произволен размер, които не притежават посочените недостатъци (преходи и повторения) е непосредственото синтезиране на текстури. Използуваните методи могат да се разделят на две групи:

Методи, които генерират текстури по зададен образец.

Методи, които генерират текстури посредством избрана случайна или детерминирана функция. В настоящата статия е предложен метод за синтезиране на текстури посредством покриване на двумерна област с геометрични фигури, получени чрез модулация на параметрите на покритие (паркетиране) с многоъгълници. Методът представлява по-нататъшно развитие на предложения в [1] метод за получаване на текстура посредством модулация на координатите на върховете на паркетиращи многоъгълници.

2. Покриване на равнината с геометрични фигури

В математиката плътното покриване на равнината с геометрични фигури се нарича паркетиране [2].Доказано е, че от правилните многоъгълници плътно покритие на равнината може да се получи само с триъгълници, четириъгълници, петоъгълници и шестоъгълници. Известни са много други покрития с най-различни фигури, в това число и изображения на птици и животни, повечето от които са създадени от художници, а не от математици. Независимо от вида на фигурата, тези покрития могат да се класифицират като получени от някаква детерминистична генерираща функция, чрез която се получават координатите на повтарящите се фигури.

Практическото използуване на текстурни покрития показва, че детерминистичните текстури се възприемат като създадени изкуствено, докато стохастичните текстури, (съставени от неповтарящи се елементи) се възприемат като реалистични, получени от съществуващ образец на повърхност.

2. Дефиниране на проблема

По правило алгоритмите за генериране на стохастични текстури извършват генериране на текстура по зададен образец, като се използува аранжиране на интензитета на пикселите в малка област и разпространяването им в съседна околност или различни методи за спектрален анализ на участъци от образеца и получаване на текстура върху зададена област със сходен спектрален състав. Като резултат получените текстури не съдържат различими обекти, а по-скоро области с перцептуално сходни характеристики. Могат да се посочат редица случаи за реални повърхности със “зърнеста” структура, върху която ясно се различават обекти със случайно изменящи се в определени граници параметри. Необходимо е разработването на алгоритми за изкуствено генериране на подобни текстури с предварително зададен размер..

3.Покриване със случайни многоъгълници.

Предлаганият алгоритъм за генериране на текстури използува разгледания в [1] алгоритъм за паркетиране на равнината със случайни многоъгълници. В алгоритъма се използува детерминистична генерираща функция за покриване на равнината със зададен правилен многоъгълник и случайна модулираща функция за преобразуване на многоъгълниците. Посредством генериращата функция (ГФ) се получават координатите на върховете на случайните многоъгълници. Координатите на тези точки могат да се представят посредством параметрични уравнения от вида
(1)

$$\begin{cases} x(n) = F_x(n) \\ y(n) = F_y(n) \end{cases}$$

където n е номерът на поредната точка. Видът на функциите $F_x(n)$ и $F_y(n)$ и схемата на свързването им определят вида на получените многоъгълници. При задаването на схемата на свързване на върховете на многоъгълниците е по-полезно координатите им да се представят като функция на две променливи n1 и n2. , $n1 \in [0,N1]$, $n2 \in [0,N2]$., като връзката между тях и параметъра n от формула 1 се изразява както следва
(2)

$$n = n1 + N1 * n2$$

където n1 и n2 са размерите (в брой отчети) на правоъгълната област D, в която са дефинирани функциите x(n1,n2) и y(n1,n2).

$$\begin{cases} x(n1,n2) = Mx(n1,n2) + Fx(n1,n2) \\ y(n1,n2) = My(n1,n2) + Fy(n1,n2) \end{cases}$$

Например за получаване на покритие с правоъгълници с размери A*B функциите x(n1,n2) и y(n1,n2) и граничните точки на отсечките L1, L2, L3 и L4 за свързването им се изразяват с формули (3) и (4) както следва

(3)

$$x(n1,n2) = A * n1$$
$$y(n1,n2) = B * n2$$

(4)

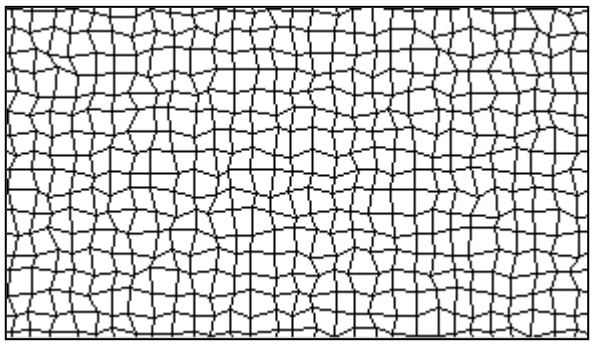
$$L1:(n1,n2) - (n1+1,n2)$$
$$L2:(n1+1,n2) - (n1+1,n2+1)$$
$$L3:(n1+1,n2+1) - (n1,n2+1)$$
$$L4:(n1,n2+1) - (n1,n2)$$

В [1] са дадени и други схеми на свързване, при които се получават покрития с друг вид правилни многоъгълници.

4. Използуване на модулация в генериращите функции.

Модулацията е термин, който се използува в теорията на сигналите. Модулация се нарича такова нелинейно преобразование на сигналите, при което един или няколко параметъра на един сигнал, наречен носещ, се променят в съответствие с друг сигнал, наречен модулиращ.[3]. В случая като дискретни сигнали се разглеждат функциите $F_x(n1,n2)$ и $F_y(n1,n2)$, чрез които се получават координатите на върховете на многоъгълниците. Модулиращите функции $M_x(n1,n2)$ и $M_y(n1,n2)$ се използуват като множители или събираеми, посредством които се получава изменение на координатите на върховете на многоъгълниците както следва:
(5)

За да не се получи след модулацията пресичане на страните на фигурите от покритието е необходимо да се подберат модулиращи функции, стойностите на които са ограничени в интервала [-D/2,+D/2] където D е разстоянието между върховете на съседните многоъгълници. Използването като модулираща функция на генератор на случайни числа дава като резултат покритие на равнината с многоъгълници, чийто размери са функция на случайна величина.
(фиг. 1)



Фиг. 1 Покриване с многоъгълници, модулирани със случайна функция.

5. Изглаждане на контурите на обектите с цифров нискочестотен филтър.

Текстурите, съставени от геометрични фигури се възприемат като изкуствено създадени. За получаване на текстура, наподобяваща естествени образци е необходимо преобразуването на геометричните фигури в обекти с гладки контури. За изглаждане на контурите на обекти може да се използува разликата в спектралния състав на параметричните функции, които съответствуват на гладки контури и спектъра на параметричните функции, които съответствуват на контурни линии, които съдържат пулса-

ции, шумове и резки изменения в направлението на контура. Локалните екстремуми на контурната функция добавят високочестотни съставящи към нейния спектър. Поради това изглаждането на контурите на обект в дискретно изображение може да се извърши посредством нискочестотна филтрация на параметричните функции на контура. Общата формула, по която се изчислява откликът y(n) на един едномерен цифров филтър се дава с израза [4].

(1)

$$y(n) = \sum_{m=-\infty}^{\infty} h(m).x(n-m)$$

където:

с h(m) е обозначена импулсната характеристика (ИХ) на цифровия филтър.

ñ x(n) - входния сигнал (входната числова редица)

В случая е удачно да се използува филтър, чиято ИХ е с крайна дължина (нерекурсивен филтър), тъй като при филтрите с безкрайна ИХ преходния процес (теоретично) е с безкрайна продължителност, а това в конкретния случай има като резултат зависимост на вида на получавания изгладен контур от началната точка на проследяване на контура на зададения обект. И така, по предлагания метод изгладените контурни функции $n1_s(n)$ и $n2_s(n)$ се получават посредством филтрация с нерекурсивен цифров филтър, като стойностите на $n1_s(n)$ и $n2_s(n)$ се изчисляват по формулите

(2)

$$\left| \begin{array}{l} n1_s(n) = \sum_{m=-N/2}^{N/2} h(m).n1(n-m) \\ n2_s(n) = \sum_{m=-N/2}^{N/2} h(m).n2(n-m) \end{array} \right.$$

където:

с n1(n) и n2(n) са обозначени параметричните функции на зададения контур

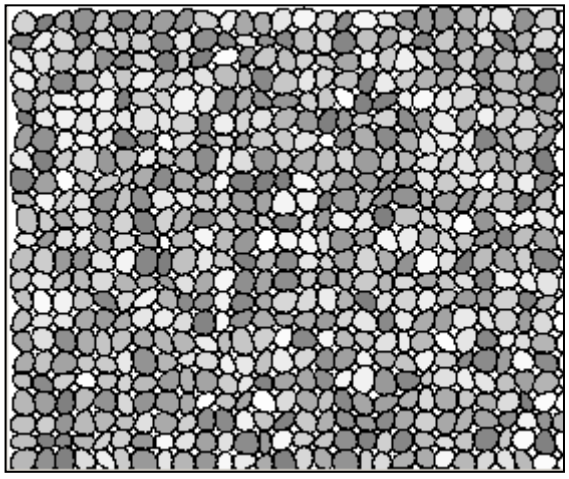
с $n1_s(n)$ и $n2_s(n)$ - параметричните функции на изгладения контур.

с h(n) - импулсната характеристика на филтрите

За филтриране може да се използува усредняващ филтър като интервалът на усредняване N се подбира от оператора. Тъй като на практика интервалът на усредняване не може да бъде много голям (най-често 30-40 отчета) то няма голяма разлика между простото усредняване с импулсна характеристика h(n)=1/N и използуването на нискочестотен филтър с някоя от известните тегловни функции (например прозорец на Хеминг или Кайзер) .Изглаждането на контури посредством линейна филтрация по зададен размер на прозореца на филтрите е разгледано в [5].

6. Практически резултати.

Реализирана е програма, която генерира текстурни изображения по описания по-горе алгоритм, като дава възможност на оператора да избира параметрите на генерираното изображение – размер на стъпката на координатната мрежа при получаване на покритието с правилни многоъгълници, размер на усредняващия прозорец, , цветове на фона, запълването и контурните линии и др. На фиг. 2 е показан резултат от изпълнението на програмата при използуване на случайна модулираща функция и усредняване с прозорец с размер 41 отчета.



Фиг. 2 Текстура, получена след изглаждане на контурите с НЧ филтър

7. Заключение

Предложеният алгоритъм дава възможност за генериране на текстури чрез модулация на параметрите на геометричните фигури, с които е извършено паркетиране на двумерна област и филтриране на параметричните функции на контурите на обектите, разглеждани като цифрови сигнали. По-нататъшно развитие на алгоритъма може да бъде получаването на покритие (паркетиране) със зададен набор от геометрични обекти, които да се редуват в случайна последователност, и които след това да бъдат подложени на модулиране и филтрация. Това би дало възможност за получаване на голям брой текстури от различни начални набори от обекти

Литература

[1] О. И. Железов, Даниела Д. Илиева, Алгоритъм за генериране на текстурни изображения, Юбилейна научна сесия ТУ-Варна 1991г.

[2] М. Гарднер, Математические развлечения, “Наука и изкуство”, С. 1980.

[3] В. Халачев, Г. Стоянов, Теория на сигналите, С. “Техника”, 1980.

[4] Л. Рабинер, Б. Гоулд, Теория и применение цифровой обработки сигналов. М. “Мир” 1978

[5] Т. Павлидис, Алгоритмы машинной графики и обработки изображений. пер. Н.Гуревич, М. “Радио и связь”, 1986г.

[6] О. Железов, Метод за изглаждане на контури. Научна сесия “Дни на науката - Варна”, 1998г

ИНТЕЛИГЕНТЕН ГРАФИЧЕН ДИСПЛЕЙ ЗА ПРОМИШЛЕНИ КОНТРОЛЕРИ

Сава Иванов

Анотация: В статията се разглежда архитектурното решение и проблемите свързани с проектирането на интелигентен графичен дисплей за промишлени контролери. Дисплеят има следните по-важни характеристики: LCD графичен дисплей, 240 x 128 точки, сензорен екран (тъчпанел), часовник за реално време, аналогов вход и изход, 6 цифрови входа и 4 цифрови изхода и канал за връзка с промишления контролер чрез интерфейс RS232/485.

INTELLIGENT GRAPHIC DISPLAY FOR PLC

Sava Ivanov

Abstract: The present article deals with the architecture solution and the problems connected with designing an intelligent graphic display for PLC. The display has the following characteristics: LCD graphic display, 240 x 128 dots, touch panel, real-time clock, analogue input and output, 6 digital inputs and 4 digital outputs and a communication channel with the interface RS232/485

.

ВЪВЕДЕНИЕ

Промишлените контролери (ПК) са специализирани компютърни системи предназначени за изграждане на автоматизирани системи за управление на технологични процеси или промишлени изделия. Един от задължителните компоненти на такава система е наличието на пулт за оператора, чрез който се въвеждат параметри на системата, избират се режими на работа, индицират се текущи състояния и стойности на параметри и др. Всяка фирма-производител на промишлени контролери предлага и гама от пултове и дисплеи за връзка със системата. Те, обаче, не винаги са подходящи за използване в конкретни задачи. Докато ПК лесно могат да се унифициат, то разнообразието от приложения изисква и разнообразие от управляващи пултове и лицеви панели на системата. Разположението на бутоните и техния брой, разположението на индикаторите, светодиодите и техния брой и т.н. е строго индивидуално за всяка задача, за всеки проект. При избор на управляващ панел желанието на проектантите е той да има следните качества:

- да може да се извежда както буквено-цифрова, така и графична информация. Дисплеят да позволява бърза и лесна промяна както на текстовата, така и на графичната информация за различни режими на работа на системата. Разра-ботчика да има възможността и сред-ствата сам да синтезира графиките и разположението им в екранното поле.
- броя, разположението и функциите на бутоните да не са фиксирани, а да могат да се определят от разработчика при дизайна на лицевия панел.
- лицевият панел да е максимално защитен от прах, влага, вибрации, единични удари и др.
- лицевият панел да е автономно интелигентно устройство, а връзката му с ПК да се осъществява по стандартен

интерфейс

- да е лесен за вграждане и обслужване, да е с висока надеждност и ниска цена.

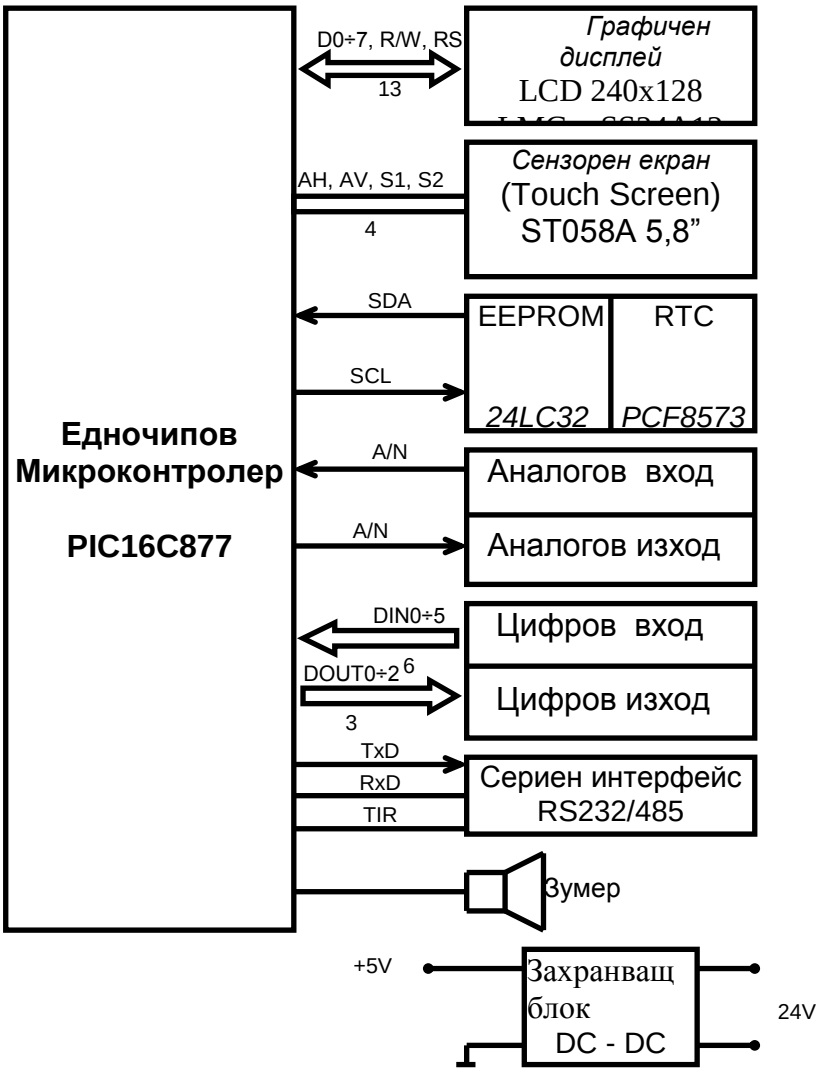
Цел на настоящата разработка е да удовлетвори в голяма степен тези понякога противоречиви изисквания като използва добри технически решения и съвременни компоненти.

2. ТЕХНИЧЕСКО РЕШЕНИЕ

При избора на техническо решение целта е била да се удовлетворят следните изисквания:

- основните функции на лицевия панел т.е. индициране на резултати, въвеждане на параметри, избор на режими и др. да удовлетворяват изискванията от т.1
- възможност за включване на външни бутони от типа ON-OFF или външна клавиатура
- лицевият панел да има възможност за управление на светлинен и звуков алармен сигнал
- лицевият панел да може да има часовник за реално време

Блоковата схема на лицевия панел е дадена на фиг.1.



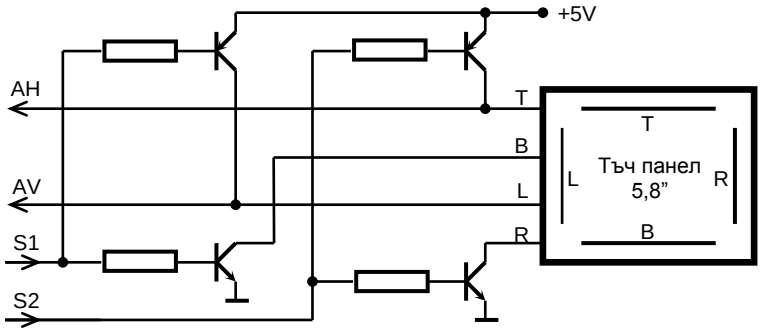
Фиг.1

Функциите на отделните блокове са следните:

- блок “графичен дисплей” – предназначен е за извеждане на буквено-цифрова или графична информация. Избран е дисплейния модул LMG-SS24A12 на фирмата SDEC [1]. Това е интелигентен модул управляващ LCD панел с 240 x 128 точки. Има вграден контролер, който поддържа осембитов паралелен интерфейс с микропроцесорната система и прави обслужването на модула изключително леко.

Модулът има 13 програмно достъпни регистъра с помощта, на които може да бъде избран режима на работа (графичен или буквено-цифров), да се определят размерите на буквите, да се управлява курсора и др. В модула има вградена VRAM, в която се зарежда информацията подлежаща на индициране. Капацитетът й позволява в нея да се пази информация за 4 графични екрана или много повече екрани, ако информацията е буквено-цифрова. По програмен път се определя началния адрес на VRAM, от който информацията се извежда върху дисплея. Като подсветка се използват светодиоди, а яркостта на светене на точките може да се регулира. блок “сензорен екран” – вместо бутонен блок с фиксирано разположение и брой бутони е избрано решение с използване на тъч-скриин (touch screen). Това е сензорен екран от резистивен тип реагиращ на докосване с пръст. Избран е 5,8” екран ST-058-01 на фирмата Suntai с размер на активното поле 121,4 x 91,2 mm [2]. Връзката му с микропроцесорната система се осъществява с помощта на четири линии -T, B, L, R (Top, Bottom, Left, Right). Панелът е от резистивен тип т.е. докосването на екрана променя омическото съпротивление между линиите съответно по хоризонтала и верикала. На фиг.2 е дадена схемата за връзка на тъч-панела с микропроцесорната система.

За позицията на докосване се съди по нивото на аналоговите сигнали AH и AV. Замерването на точката по двете оси на координатата се извършва последователно. При s_1 =0 и s_2 = 1 се замерва AH (хоризонталата), а при s_1 = 1 и s_2 = 0 се замерва AV (вертикалата). При използване на 8 битови АЦП размера на екрана е 256 x 256 точки. Времето за установяване на точната стойност на съпротивлението е 15 ms след докосването. Екрана е напълно прозрачен и допуска до 100 милиона докосвания със сила от 20 до 80 g.



Фиг.2

- блок EEPROM – служи за съхранение на параметри на системата, информация за графичните екрани и разположението на бутоните върху сензорния екран. EEPROM се свързва с микропроцесорната система по стандартен интерфейс I^2C.

- блок RTC (Real Time Clock) – служи да отчита астрономическото време. Избран е чипа PCF8573 на фирмата Philips, който отчита минути, часове, дати, дни от седмицата.

Генераторът му е кварцово стабилизиран, а захранването му е акумулаторно, което го прави енергонезависим. В него е вградена и функция “аларма”. Настройката на часовника и четенето на данните се осъществява по стандартен интерфейс I^2C.

- блок “цифрови входове” – това са 6 еднотипни схеми осигуряващи опторазделяне на източника на сигнала с микропроцесорната система. Входният сигнал е стандартен- 24V/ 10mA и обикновено се генерира от бутони или ключове тип ON – OFF. С успех към тези входове могат да бъдат подключени и индуктивни, оптически или други p –n – p или n – p – n датчици.
- блок “цифрови изходи” – това са 3 еднотипни схеми на опторазделени транзисторни изходи с параметри на изходния сигнал 24V/ 1A. Изходите са предназначени за управление на светлинна или звукова аларма, но могат да се използват за управление и на изпълнителни механизми от типа ON – OFF.
- блок “сериен интерфейс RS232/485” – осигурява връзка на инте-лигентния дисплей с промишления контролер. Избора на интерфейса RS232 или RS485 се уточнява още при монтажа на елементите.
- блок “зумер” – основното му предназначение е да издава звуков сигнал при докосване на тъч-панела. Може да бъде използван и като сигнал с друго звучене при неправилно въведени параметри и др.
- блок “аналогов вход и изход” – това са блокове, които не са свързани с основните функции на интелигентния дисплей, но разширяват неговите функции и възможности за приложения. Аналоговият вход обработва информацията от температурен датчик Pt100, а аналоговият изход генерира изходен сигнал от 0 до 5 V.
- блок “едночипов микроконтролер”. Избран е ЕМК PIC16F877 на фирмата Microchip [3,4]. Вграденият 8 битов АЦП се използва за обслужване на тъч-панела и аналоговия вход. Таймерният блок се използва за генериране на ШИМ, от който пък се получава аналоговия изход. Използват се и вградените контролери за асинхронен (RS232) и синхронен (I^2C) интерфейс. Останалите изводи на ЕМК се използват като стандартни входове или изходи и обслужват останалите блокове от системата.

3. ЗАКЛЮЧЕНИЕ

Интелигентният графичен дисплей е проектиран, изработен, настроен и внедрен в едно приложение. За да се постигне по-голяма гъвкавост и удобство при проектирането на приложното програмно осигуряване предстои създаването на програма за PC с помощта, на която ще се създават макроси за графичните екрани, макроси за докосването на тъч-панела и цировите входове-изходи. Тази информация предварително ще бъде заредена в EEPROM-а и по този начин ще се облекчи комуникацията по интерфейса RS232/485.

ЛИТЕРАТУРА

1. SDEC Technology Corporation, Dot Matrix Liquid Crystal Display Module LMG – SS24A12 Serial, User Manual
2. Suntai International Co. Ltd., Touch screen, **www.sun-tai.com.tm**
3. Ц.Василев, А.Смрикаров, Специализирани микрокомпютърни системи, част I, Русе 1997г.
4. PIC16F84, Data Sheet, Microchip 1998

5.

Множества на Жюли за трансцендентни изображения

Слава М. Йорданова, Николай Т. Костов

Резюме: В статията се разглеждат проблемите и методите на фрактално генериране на можества на Жюли за трансцедентни изображения. Дадени са методи за фрактално построяване на множествата на Жюли. Генерирани са определени примери и е реализирана програма на C++.

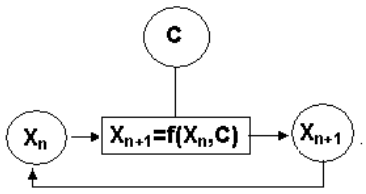
July's Multitudes for transcendental images

Slava M. Yordanova, Nikolay T. Kostov

Abstract: In this issue are shown the problems and the methods of fractal generation of July's multitudes for transcendental images. Methods for fractal building of July's multitudes are presented. Certain Examples are generated and is realized a program written in C++.

1.Увод

Компютърната графика е в период на интензивно развитие. Рисунките представляват процес изразяващ идеализираната действителност. Не съществува реална структура, която последователно увеличена, безкраен брой пъти да изглежда неизменна. Принципът на самоподобие в приближен вид съществува и в природата: в линията на брега на морето или реката, в очертанията на облаците и дърветата и в йерархическата организация на живите системи. Тази фрактална геометия на природата е открита от Б. Манделброт. Процесите, пораждащи такива структури, отдавна се изучават от математиката и физиката. Това са обикновенните процеси с обратна връзка. [1][2][4]



Фиг.1

Необходимо е да съществува нелинейна зависимост между резултата и началната стойност.

Фиг.1 показва, че правилото $x \rightarrow f(x)$, зависи от параметъра "c". Ако се започне итерационният процес с някаква произволна стойност x_0, то като резултат ще се получат последователности от стойности x_1, x_2, ...x_n, Целта на изследването е: представеният по-горе процес да се изследва вместо с реални числа (както е разработката на Манделброт) - с комплексни числа и процеса да се разглежда не върху права а в равнина. Такава структура се нарича множество на Жюли. Множествата на Жюли представляват фрактални структури по определението на Манделброт и имат сложна динамика.[4] За целта е създадена и компютърна програма на езика C++ за реализация на разглежданата теория и предлаганите методи за генериране на множества на Жюли.

2. Математическо представяне на множествата на Жюли за трансцедентни изображения.

В статията се разглеждат множества на Жюли в комплексна плоскост. Сравняват се някои свойства на тези множества, получени, чрез използване метода на Нютон за вещественi уравнения.[2][3]

Трансцендентните изображения, се описват като:

$$E_\lambda(x)=\lambda\exp(x),$$
(1) $$S_\lambda(x)=\lambda\sin(x),$$
$$\lambda\in C \text{ в } \overline{C} \quad .$$

Определяме множеството на Жюли за E_λ (или S_λ) в съответствие с (1).

Наблюдават се съществени изменения на поведението на E_λ, когато λ движейки се по действителната ос, преминава през значението $1/e$.[2][4]

При $\lambda<1/e$ е налице притегляща неподвижна точка Q_λ и отблъскваща неподвижна точка P_λ, докато при $\lambda>1/e$ изобщо няма неподвижни точки. Очевидно, че при $\lambda<1/e$ и

(2) $$\{x\in R: x>P_\lambda\}\subset J_\lambda.$$

Този лъч се нарича косъмче и играе решаваща роля при описването на J_λ.

- Ако $\lambda>1/e$, то $J_\lambda=C$.
- Ако $\lambda<1/e$, то J_λ представлява не плътно множество от криви, образуващи границите на една единствена област на притегляне.

По такъв начин, когато λ расте и преминава през значението $1/e$, множеството J_λ се разпада. Изображения на множествата на Жюли за подобно семейство се получават трудно.[3]

Вътрешната област на множеството $\mathbf{C}$ съдържа компоненти, на които отговарят периодите на съответните притеглящи периодични точки.

(3) Ако $E_\lambda^n(0)\to\infty$, то $J_\lambda=\mathbf{C}$.

На основата на това твърдение точката в λ-плоскостта може да се оцвети в черен цвят, когато при известно $n<N_{\max}$ се изпълнява неравенството $E_\lambda^n(0)\geq: M$, където $M>>1$. Например бялото множество съдържа такива точки, като $\lambda=2k\pi i$, за които $E_\lambda^n(0)=E_\lambda(\lambda)=\lambda$. В общия случай, ако точка 0 се явява периодична (т. е. $E_\lambda^n(0)$ се явява периодична при известно $n\geq1$), то $J_\lambda=C$.

За всяко λ, за което $\left|E_\lambda^n(0)\right|\geq M$ при известно $n<N_{\max}$, съществува такова най-малко n. С него е свързан приписваният на λ индекс, на който може да се съпостави определен цвят в таблицата на цветовете, за да се построят цветни изображения.[3]

3. Построяване на изображения на множествата на Жюли

3.1 Метод на обратното итериране (МОИ)

Ако е зададено рационално изображение R и е известна една периодична отблъскваща точка $\overline{x}\in J_R$, то може да се изчисли:

$$J_R^n=\left\{x\in\mathbf{C}: R^k(x)=\overline{x} \text{ при известно } k\leq n\right\}$$

Понеже J_R е затваряне на $\left(\bigcup_{n\geq0}J_R^n\right)$, може да се очаква, че изображението J_R^n при достатъчно голямо n ще бъде достатъчно близко п до изображението J_R. Действително в този случай, когато J_R^n се разпределя равномерно по J_R, този метод дава напълно удовлетворително изображение J_R. Казано по-просто, ние наричаме отрицателната полутраектория $Or^-(\overline{x})$ равномерно разпределена, ако броя на

точките в $J_R^n \cap D(x,\ \varepsilon)$, където $(x \in J_R,\ \varepsilon > 0$ и $D(x,\ \varepsilon) = \{y : |y - x| < \varepsilon\})$ по същество не зависи от x при големи n. За съжаление такава ситуация е нетипична. Обикновенно J_R има околности, посещавани крайно рядко. В този случай прекия МОИ не работи. Напомняме, че броят на елементите на множеството J_R^n расте както d^n, където d – е степента на R. Типичната ситуация е показана на фиг.2.[3][4]

В направените експерименти J_R се налага върху квадратна решетка в $\mathbf{C}$. Броят на точките от J_R^n във всяка малка клетка от решетката се изчислява и се изобразява във вид на вертикална линия, в резултат на което разпределението се визуализира. Очевидно, най-често се посещават "краищата" на множеството J_R, докато точките на разклонение, както се оказва, по-често се избягват. Въпреки това при $n = 21$ се образува достатъчно плътно множество от точки, така че неравномерностите почти не оказват влияние.

Като допълнение към тези недостатъци, МОИ не може да се използва и без подходящо управление на базата данни, поради бързото нарастване на d^n.[3]

Има много начини да се номерират различните нива на $J_R^k,\ k = 0,\ 1,\ \ldots,\ n$. Пряката номерация по редове води до това, че за изчисление на $(n+1)^{\text{вото}}$ ниво ще са необходими всичките d^n елемента на множеството J_R^n, които бързо ще запълнят целия достъпен обем на машинната памет. Обаче ако се сметне, че n итерации ще се окажат достатъчни, то може да се използва очевидния метод за номерация на дървото, при който ще са необходими само $d \cdot (n-1)$ единици памет за построяване на J_R^n.

В общия случай разпределението $Or^-(\bar{x})$ се характеризира с определена мярка μ_R, носител на която се явява J_R: [3], Съществува инвариантна борелевска мярка μ_R с носител J_R.

Каквото и да е $x \in \bar{\mathbf{C}} \setminus E$ (E - е определено изключително множество) μ_R се явява слаба граница на мярката $\mu_{m,\ x}$, където

$$\mu_{m,\ x} = \frac{1}{d^m} \sum_{\zeta \in R^{-m}(x)} \delta_\zeta \quad \text{и} \tag{4}$$

$$\delta_\zeta(u) = \begin{cases} 1, & \text{ако } u = \zeta \\ 0, & \text{ако } u \neq \zeta \end{cases}$$

Ако μ_R беше известна предварително, то би могло да се модифицира МОИ. Приведената по-долу стратегия с известно приближение отчита мярката μ_R, макар на практика тя да е неизвестна.[3][4]

3.2 Модифициран метод на обратното итериране (ММОИ)

Точките от $J_R^k, k < n$ принадлежащи на участъците с голямо тегло спрямо μ_R, следва при обратната итерация да се използват значително по-рядко, отколкото се срещат. В действителност броят на използваните в итерационния процес точки трябва да бъде обратно пропорционален на μ_R. Стратегията е следната: върху J_R се налага квадратна решетка с малък размер β. След това, за всяка клетка B от решетката следва да се прекрати използването от нея на точки за обратно итериране, ако определен брой $N_{\max}$ точки в B вече е използван. Естествено, че оптималният избор на β и $N_{\max}$ зависи от μ_R и от параметрите на компютърната графика, като разделителната способност на наличната система. Следователно се изискват интерактивни и адаптивни алгоритми. [3]

3.3 Метод на сканиране на границите (МСГ)

Ако R има притегляща неподвижна точка a, т.е. $R(a)=a, |R'(a)|<1$, и ако $A(a)$ – е нейния басейн на притегляне, то:

(5) $$J_R = \partial A(a)$$

Ако R има повече от една притегляща неподвижна точка, да кажем a и b, то е налице достатъчно прост начин да се получи изображението на множеството:

(6) $$\partial A(a) = J_R = \partial A(b).$$

Избира се квадратна решетка, покриваща известна област в $\overline{C}$, в която трябва да се получи J_R. Нека B – е произволна отворена клетка с размер β. Предполагаме, че $B \bigcap J_R \neq \varnothing$. Тогава B трябва да съдържа точки както от $A(a)$, така и от $A(b)$.[3]

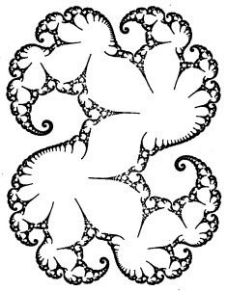
На това е основан следният алгоритъм: нека $N_{\max}$ и $0<\varepsilon<1$ са зададени и нека B – е произволна клетка от решетката с върхове c_1, c_2, c_3 и c_4. Трябва да се установи:

- принадлежат ли всички върхове на един и същи басейн или върховете принадлежат на различни басейни. Във втория случай B се оцветява, например в черен цвят, а в първия – с бял. За това трябва числено да се определи, принадлежи ли например c_i на $A(a)$ или не.
- ако в някой случай $k \le N_{\max}$ се окаже, че $\left|R^k(c_i)-a\right|<\varepsilon_r$, то $c_i \in A(a)$, Ако не, ще считаме, че c_i не принадлежи на $A(a)$

Параметрите β, ε и $N_{\max}$ отново силно зависят от R, и може да се окаже, че не могат да бъдат избрани така, че да се получи удовлетворително изображение J_R. Изискването, да съществуват най-малко две притеглящи неподвижни точки може да не е така строго, ако е налице известна допълнителна информация за изображението R. Така например, може да се сравни басейна на притегляне с диска на Зигел, пръстена на Ерман или даже с множеството, при което R е разходим.

4. Заключение

Опитът показва, че построяването на множеството на Жюли при наличие на параболични точки е трудна задача.



фиг.2

На фиг.2 е показан резултат, получен с помощта на отрегулиран по подходящ начин алгоритъм МСГ.

- Първо, ние знаем, че $x_0 = 0 \in J_{R_0}$
- Второ, понеже $\lambda^{20} = 1$, множеството на Жюли трябва да се приближава към точка $x_0 = 0$ от двадесет различни направления (между 20 листчета).

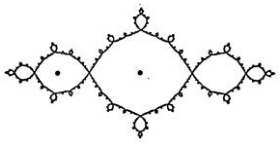
В действителност на рисунката се вижда само намек за тези направления, но множеството на Жюли явно изглежда отдалечено на известно разстояние от x_0. Този ефект по своята природа се явява изчислителен, т. е. той е напълно различен от ефектите, породени от вътрешната неравномерност на инвариантната мярка.

Нека единицата за грешка от закръгление на дадения компютър да бъде eps (т. е. eps – е най-голямото машинно число, за което $1+eps=1$ при аритметиката на машината). По-нататък, нека R удовлетворява (6).

(7) $h.R.h^{-1}(x)=\lambda x(1+x^{kn})$.

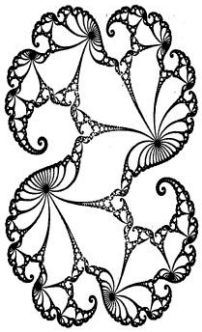
Тогава, ако $|x|<eps^{\frac{1}{kn}}$, то очевидно, е невъзможно да се отличи λx от $\lambda x(1+x^{kn})$ и следователно $R^n(h^{-1}(x))$ не може да се отличи от $h^{-1}(x)$.

С това се обясняват не само недостатъците (фиг.2) ($kn=20$), но и това, че този ефект почти не се забелязва на фиг.3 ($kn=2$).



фиг.3

На фиг.4 е показано как трябва да изглежда изображението от фиг.2.



фиг.4

Изображението е получено чрез алгоритъма ММОИ и приложен към

(8) $R_\varepsilon(x)=(1+\varepsilon)\lambda x+x^2, \lambda=\exp\left(\frac{2\pi i}{20}\right)$

При изменение на ε от 0 до известно малко положително число параболичната неподвижна точка бифуцира в притеглящ цикъл, на периода 20, а x_0 става отблъскваща неподвижна точка.

Методът за компресия на сигнали и изображения, чрез фракталното им представяне е един от най-перспективните и интересните в тази област. Той може да бъде обект и на допълнителни научни изследвания, с цел създаване на нови, реални апаратно-програмни средства за масови приложения в ефективни и евтини компютърни мултимедийни системи. Например за целите на телевизията с висока точност - HDTV подобни резултати ще бъдат твърде необходими, поради несъмненото й бъдеще и приложение в глобален план.

Литература

1. Сл. Йорданова, Б. Рачев, В. Наумов, Мултимедия и компресия на изображения, София 2000г.
2. Х. Пайтен, П.Х.Рихтер, Красота фракталов, Издателство “Мир” 1993г.
3.] Davis P.J. Hersh R. The Mathematical Experience. Birkhauser, Boston
4. Г. Шустер, Детерминированнъй хаос, Издателство “Мир” 1999г.

Генетични алгоритми и създаване на разписи

Милена Н. Карова

Резюме: В доклада се разглежда използването на един по- малко разпространен метод за решаване на оптимизационни задачи Генетични алгоритми (ГА). Накратко е разгледана неговата същност, действие и приложения. Предложена е една много интересна структура на ГА за решаване на проблема с разписите: паралелен ГА. Разгледани са неговите характеристики, използването на основните генетични операции: инициализация, селекция, кросовър, и мутация.

Timetabling using Genetic Algorithms

Milena N. Karova

Abstract: The report examine the using of a method, less popular that solves NP problems: Genetic Algorithms. It is more shortly considered its nature, application and operating. One very interesting structure GA for the decision a problem with timetable is offered: parallel GA. It examines its characteristics, based genetic operation's using: initialization, selection, crossover and mutation.

1. **Увод**

Навсякъде по света, от традиционата оптимизация в проектирането до такива нетрадиционни области като финансово предсказване, създаването на лекарства и др Генетичните Алгоритми (ГА) привличат внимание и решават трудоемки проблеми. Те са едни от най-популярните в сегашно време способи за решаване на задачите за оптимизация., като в тяхната основа лежи използването на еволюционите принципи за търсене на оптималното решение. На базата на Генетичните Алгоритми се изграждат програми, които трябва да решат такива трудоемки проблеми като създаването на разписанията в училищата и други институции, където това не може да се извърши без помощта на човешки ресурс. Проблемът с разписанията е вид проблем в който събитията (например изпити. класове и пр.) трябва да бъдат подредени подходящо във времето съгласно различни ограничения. Нуждата от силен метод за решаването на проблема е ясна като се има в предвид простият факт, че ако означим с "e" изпитите подходящи за "t" времеви интервал, това са t^e възможни за избор разписания, като оптималният вариант е според ограниченията на проблема [1]. Общоприетите компютърно базирани матоди за разписания се отнасят повече към просто откиване на най-късото разписание удоблетворяващо всички ограничения. Това изпълнява поставените условия, но не е напълно достатъчно за постигането на най-добрият краен резултат, което може да се постигне използваики Генетичните Алгоритми. Много често хората прибягват до използването на ръчни средства за решаване на проблема и правят редица изменения обслужващи промяната на условията. Това може да доведе до множество както значими така и финансови проблеми и нарушаване първоначалната цялостност на задачата. Изграждането на разписания спада към проблемите за оптимизация, като от тях почти всички се включват

към задачите с NP сложност (nondeterministic polynomial). Използването на детерминирани алгоритъм за тази задача би бил доста бавно и затова е необходим недерминиран алгоритм , какъвто е генетичният. Генетичните Алгоритми извършват многоточково търсене в пространството на проблема. Те осигуряват "здравина в случай, че търсеният път е в локалният минимум, което няма да означава че целият алгоритъм не е успял, а само че може да се дадат няколко оптимални решения на проблема от които потребителят ще може да избира.

2. **Генетични Алгоритми**.

Генетичните Алгоритми са най-често обща група от методи наричана еволюционно изчисление които обединяват различни варианти на използване на еволюционните принципи за достигане на поставените цели. Като причина за възникването на тези методи са станали няколко открития в биологията. Първо Чарлз Дарвин публикува през 1859 г. своята знаменита работа "Произход на видовете" където са провъзгласени основните принципи на еволюционната теория: наследственост, променливост и естествен подбор. Целта на индивида е оцеляване в постоянно променяща се среда чрез трите процеса мутация, онаследяване и селекция (естесвен подбор). Мутацията възниква тогава когато поведението на идивида се промени под въздействието на случайни външни фактори, като се променят шансовете за оцеляване на индивида предаден и в неговото поколение. Дали мутацията е успешна се определя в процеса на естествен подбор, като по-добрите оцеляват и създават свое поколение т.е. “оцелява” само най-доброто [5].

Този механизъм на действие може да се симулира в изчислителната среда за целите на машиното самообучение. При компютърно генериран модел се създават множество от кандидати, но “оцелява” само най-доброто решение. Друго възможно решение е да се запази цялата група от възможни решения , като предимството на търсене с повече от един индивид се отнася до способността за намиране на глобален максимум.

Генетичните алгоритми са свободно базирани на идеите лежащи в еволюционата теория за естесвената селекция и сегашните знания за основите на генетиката. В много източници Джон Холанд се нарича родител на съвременната теория на Генетичните Алгоритми. Негово е предложението видът на подържаните алгоритми да е точно определен – двоични числа с фиксирана дължина. Тези числа (последователности от битове) се наричат хромозоми, а битовете в тази последователност алели. Всеки хромозом поражда един индивид, като всеки алел се проявява в индивида като някаква черта, която може да е различно полезна за оцеляването. Процесът на еволюцията представлява подбор на алели избрани чрез тестване.

При генетичните алгоритми всяка хромозома кодира едно решение. Търсенето на това решение може да се сведе до намирането на екстремум на функция. Също така възможните решения на даден проблем може да не бъдат характеризирани по време на изпълнение на алгоритъма. Следователно използването на аналитично решение би било твърде сложна задача. Например задачата за търговският пътник, който трябва да посети определен брой градове по най-краткият път с използваното на аналитичен (детерминиран) алгоритъм би било доста бавно в сравнение с използването на недетерминиран алгоритъм осигуряващ по-бързо решение дори то да не е възможно най-доброто.

Генетичните алгоритми спадат към методите за оптимизация, но има три основни разлики:

-ГА работи с кодиране на параметрите, а не със самите тях.

-ГА търси като популация набор от положения, а не само едно положение.

-ГА използва само крайна информация, не извлечени или други помощни знания [5].

На фиг.1 е показана схема на прост генетичен алготиръм.

Фиг.1

3. Задачата "Създаване на разпис"

На базата на генетичните алгоритми се изграждат програми, които трябва да решат такива трудоемки проблеми като изграждането на разписанията в училищата и други институции, където това не може да се извърши без помощта на човешки ресурс. Самият проблем с разписанията е вид проблем в който събитията трябва да бъдат подредени подходящо във времето съгласно различни ограничения и затова възниква необходимостта от силен метод за решаване на големи проблеми. Именно тук прилагането на генетичните алгоритми дава задоволителни резултати. Логичният въпрос който можем да си зададем тук е защо трябва да използваме генетични алгоритми тук, когато съществуват множество други начини, други алгоритми за решаване на тези проблеми. Отговорът на този въпрос е че разбира се както във много други проблеми и тук винаги съществува повече от едно възможно решение на проблема, но има множество проблеми чиито възможни решения се характеризират с така нареченото пространство на решенията. Това пространство на решенията може да не бъде напълно известно по време на работата на алгоритъма. Тогава решаването на проблем от подобен вид чрез използването на аналитично решение (детерминиран алгоритъм) би било една твърде сложна задача като не винаги ще се получи достатъчно задоволителен резултат или за достатъчен определен от нас интервал от време.

Какво предлагат генетичните алгоритми ? При тях всяка една хромозома кодира едно от решенията и търсенето на решение може да се сведе до намирането на екстремум на функция. В процеса на еволюция използвайки различните генетични оператори – кръстосване, мутация и пр. се намира най-добрата популация.

Разглеждайки разписанията ние трябва да дадем точна дефиниция какво представляват те и с какво се свързват. Думата разписание се свързва със планиране и организиране на действия:.учебен разпис, разпис за деня, разпис на дежурствата, разписание на транспортен трафик, разпис на изпити и др.

За прости разписания, графици обхващи само малък брой хора зависещи от малък брой условия които трябва да бъдат спазени, изготвянето на разписание би било проста задача която с не голямо усилие би мога да се свърши от група от хора. За сложни разписания със променящи се условия за кратък период от време (което може да бъде от дни до месеци) очевидно такъв подход не би бил подходящ, а и полученото решение надали би задоволили нашите очаквания. Изграждането на разписания спада към проблемите за оптимизация, като от тях почти всички се включват към задачите с NP сложност.

Традиционните, две измерения на представяне на разписанията, обикновено се използват по-време на процеса на ръчно подреждане на разписанията. При тях имаме раполагане по хоризонталната ос на различните класове, а по вертикалната времевите периоди за лекциите. Тогава ще имаме за матрицата точките (i,,j) съдържаща учителите които 'дават' лекции на клас i за времевият период j. Има и друго представяне на разписанията където времевият период е по хоризонталната ос, а учителите са по вертикалната ос. Тогава точките (i,,j) на матрицата съдържат тези класове които имат лекции във времевият период j провеждани от учителите i.

Първото представяне е ефикасно когато се извършва търсене базирано на класовете, докато второто е по-подходящо при търсене базирано на учителите. В практиката се използват и двата вида представяния. И двете представениия имат преимущества които ги предпазват от скритите ограничения: В първият случай се появява сбъскване между класовете, а във вторият случай 'сблъсъкът' е между учители. За съжаление тези две представяния не подържат разделяне на лекциите, защото те въобще не правят разлика между понятията учител и предмет. По този начин ситуацията където класът има лекции водени от двама преподавателя по чужди езици е аналогично на ситуацията където класът има лекции водени от учителя по математика и по изобразително изкуство. Виждаме че първият случай е възможен докато вторият определено не.

Разликата между учител и предмет е също необходимо в следните случай: учителят дава няколко лекции по различни предмети в един и същ клас като е проверено еднаквото разпределение на лекции.

Тъи като проблемът е твърде сложен класическите алгоритми са малко или въобще неефективни, но въпреки това те дават началната база от знания за алгоритмите на евристично търсене и модифицирано отстъпване. Словашката група aSc предлага използването на софтуер за подреждане на разписанията, който използва софтуер за директно търсене. Ако по време на процеса на подреждане се взема броят на условията на твърде сложен проблем може да се прехвърли управляемата област и тогава резултатът от алгоритъма за директното търсене може да не бъде достатъчно задоволителен за допустимото време. Алгоритъмът за симулативно закаляване за решаване на проблемът с разписанията е предложен от А. Амбарсан. Табу търсенето е представено като възможно решение на проблемът с разписанията от Херц и Де Вера. За съжаление те не са достатъчно ефективни за проблем с голям интервал от решения [7]. Симулативното закаляване и табу търсенето са проучени от Колорни, Дориго и Маниецо в *Computational Opti-mizationand Applications Journal*.

3. Паралелен Генетичен Алгоритъм за решаване проблема с училищни разписания.

Проблемът за създаване на правилно разписание включва планиране на класове, ,учители и стаи във фиксирани по брой периоди, по такъв начин че никакъв учител, клас или стая да не се използват повече от един път за периода. Тук под правилното разписание се разбира, класът да може да се планира паралелно с всеки друг клас. През целия период класът изучава предмети. Конкретната комбинация - учител, предмет, стая и клас тук се нарича кортеж. Кортежът може да се използва нееднократно за седмица. По този начин проблема разписание може да бъде разгледан като

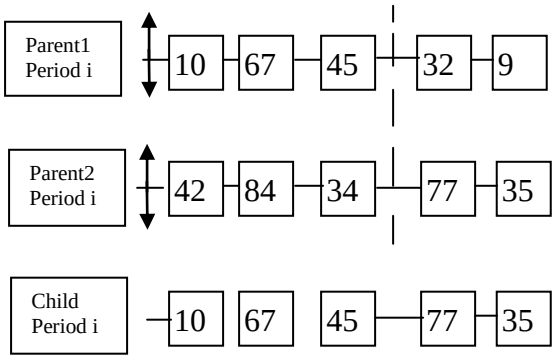
планиране на множества кортежи така че учителят класа или стаята да не се появяват повече от веднъж за периода. Определя се фитнес функцията (оценъчна функция) за всяко разписание. Възможно е да дефенираме обективна функция за оценяване на дадено разписание. Тази функция е мярка за качеството на решение. Приемливото разписание има стойност за тази функция 0. Следователно проблемът за оптимизацията се свежда до намаляване стойността на тази функция. Функция за всеки период може да бъде изразена като сума на три компонента, съответстващи на стойността на класа, стойността на преподавателя и стойността на стаята. Не е небходимо строго да се сумират компонентите, ако те могат да се обединят за да отразят качествено решение. По този начин процесът на оптимизация може да се управлява към такова решение, в което някои типа ограничения се явяват по-важни от други. Ако класът не се появява през цялото време или се появява само веднъж в течение на периода, тогава стойността на този клас ще бъде нулева. Стойността на класа за периода - е сумата от всички стойности на класа. Така всяка методика за оптимизация трябва да намери конфигурация с най-ниска възможна стойност.

Разписанието може да се представи с фиксиран набор от кортежи които съдържат номер на класа, номер на учителя и номер на стаята. При планирането на алгоритъма трябва да се определи номер на периода във всеки от тези кортежи така че дадения клас, учител или стая да не се появяват повече от 1 път в течение на периода. За да се използват ГА зарешаване на проблема е необходимо да се разработи представяне на разписание което да се отразява в хромозомите. Всяка хромозома е сформирана от гени които представят някакво свойство на разписание. Фиг.2 показва просто разписание като колекция от кортежи.

L 1	L 2	L 3	L 4		L n
C 4 T 5 R 12	C 1 T 2 R 4	C 9 T 5 R 11	C 4 T 9 R 8	●●●●	C 3 T 3 R 5

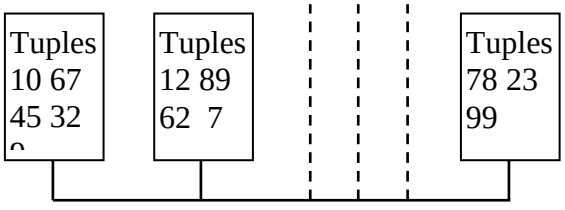
Фиг.2

На Фиг.3 са показани много кортежи опаковани във всеки период.



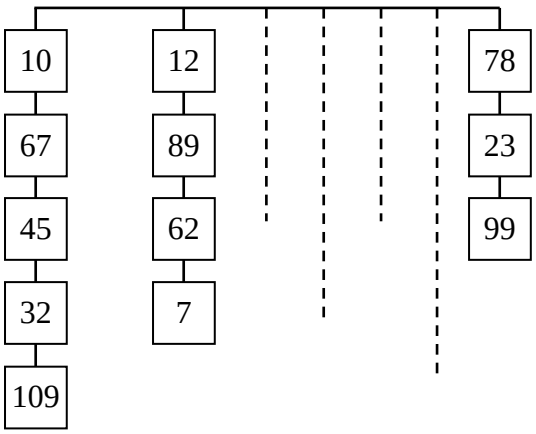
Фиг.3

Оценъчната функция на разписанието може да бъде изчислена със добавянето на броя на сблъскванията във всеки период. Хубавите и лошите атрибути на разписанието са свързани с качеството на опакованите кортежи в периода. Фиг.4 показва едно възможно разпределение на кортеж на хромозоми. Тук всеки период представлява хромозома. Използвайки това разпределение се позволява добрите опакования да се запазят между поколенията, защото съответстващите им периоди са съвместими.



Фиг.4

Прилагат се съответно процесите на кръстосване и мутация за въвеждане на разнообразие в поколението и разрешаване на новия клас, учител и разпределението на стаите да се развива. Фиг.5 показва процеса на сливане на две разписания за да се направи ново. Алгоритъмът е разгледанв перспективата на поколението, и позволява случаен избор на родители.



Фиг.5

Сливането е изпълнено на произволно избрано място на кръстосване в границите на периода и е направено в течение на всеки период на индивида. Мутацията е изпълнена на произволно избрани периоди и кортежи и става с някаква определена вероятност.

4. Заключение

Проблемът с разписанията, всяка година в училищните институции се решава с помощта на човешки ресурс. Навлизането на Генетичните

Алгоритми и увеличаващият се интерес към тях ще спомогне за оптимизирането на проблемите с разписанията, намаляване на трудоемката работа и автоматизиране и. Възникването на нови въпроси и проблеми, на които да се търси решение ще спомогне за глобалната популяризация на Генетичните Алгоритми и тяхното бурно развитие.

5. Литература

[1]. de Werra, D., "An Introduction to Timetabling", European Journal of Operational Research, Vol. 19, pag.151-162.

[2]. Hertz A., de Werra D., "Informatique et horaires scolaires", Output, Vol.12, pag.53-56.

[3]. Goldberg D. L., "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley, 1989

[4]. Goldberg D, "Web Courses", http://www.engr.uiuc.edu/OCEE, 2000.

[5]. Mitchell M., "An Introduction to Genetic Algorithms", Massachusetts Institute of Technology, 1996.

[6]. Rich, D. C. “A Smart Genetic Algorithm for University Timetabling”, In Lecture Notes in Computer Science, Vol.1153, p181-197, Springer.

[7]. Paechter B., Rankin R., Cumming A., "Timetabling the Classes of an Entire University with an Evolutionary Algorithm", Napier University, Edinburgh, Scotland.

Характеристики и приложения на турбо кодовете

Николай Т. Костов, Слава М. Йорданова

Резюме: Турбо кодовете са нов клас от мощни кодове за корекция на грешки. В статията се разглеждат основните принципи за кодиране и декодиране на турбо кодове. Представен е опростен математичен апарат за изследване на асимптотичното поведение на турбо кодове. Анализиран е широк кръг фактори, които влияят върху вероятностните характеристики и приложенията на турбо кодовете.

Characteristics and applications of turbo codes

Nikolay T. Kostov, Slava M. Yordanova

Abstract: Turbo codes are a new class of powerful error correction codes. In this paper, an overview of basic principles of turbo encoding end decoding is presented. A simplified mathematical framework for investigating the asymptotic performance of turbo codes is given. A wide range of factors that influence the error rate performance and applications of turbo codes is analyzed.

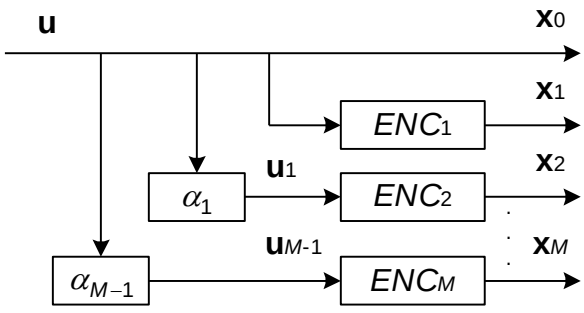
1. Увод

През 1993 г. Беру и др. [1] предлагат нов клас от мощни кодове за корекция на грешки, наречени турбо кодове. Симулацията на турбо код със скорост 1/2 по комуникационен канал с адитивен бял Гаусов шум показва, че вероятност за грешка на бит $P_b = 10^{-5}$ се достига при отношение сигнал/шум 0.7 dB. Значимостта на постигнатият резултат може да се интерпретира лесно, като се знае теоретичната граница на Шенон [2] от 0 dB за код със скорост 1/2. От друга страна, посочената стойност на вероятността за грешка на бит е постигната при използване на информационен блок с дължина 65536 бита и декодиране с 18 итерации на блок. В следствие на голямото закъснение при кодирането и декодирането, оригиналният турбо код [1] има ограничено приложение и не може да се използва за предаване на аудио и видеоинформация в реално време.

Настоящата статия има за цел да разкрие основните принципи за кодиране и декодиране на турбо кодове в контекста на факторите, които определят характеристиките и приложенията на тези кодове.

2. Кодиране и декодиране на турбо кодове.

Обобщена структурна схема на турбо кодер е показана на Фиг.1.



Фиг.1. Обобщена структурна схема на турбо кодер.

Типичната структура на турбо кодер включва паралелно свързване на $M \geq 2$ рекурсивни систематични конволюционни кодера с използване на $M-1$ блока за псевдослучайна пермутация. На Фиг.1. информационната последователност $\mathbf{u}$ с дължина k бита се кодира от рекурсивните систематичните конволюционни кодери ENC_1, ENC_2,…, ENC_M. Първият кодер ENC_1 оперира непосредствено с информационната последователност $\mathbf{u}$, а кодерите ENC_2,…, ENC_M оперират с нейните пермутирани версии $\mathbf{u}_1,...,\mathbf{u}_{M-1}$. Пермутирането на информационната последователност (т.е. промяна на наредбата на информационните битове без повторение) се изпълнява посредством законите за пермутация $\alpha_1,...,\alpha_{M-1}$, които (типично) са псевдослучайни и априорно известни на декодера. При използване на еднакви компонентни кодове кодираната последователност на изхода на турбо кодера се състои от систематичната част $\mathbf{x}_0 = \mathbf{u}$ и контролните последователности $\mathbf{x}_1 = \mathbf{u} \cdot g_1 / g_0,\ldots,\mathbf{x}_M = \mathbf{u}_{M-1} \cdot g_1 / g_0$, където g_0 и g_1 са генераторните полиноми на конволюционните кодери. По този начин скоростта на турбо кода е R = 1/(M+1). В практиката най-често се използва паралелно свързване на два конволюционни кодера, при което скоростта на турбо кода е R = 1/3. Повишаване на скоростта на турбо кода е възможно с използване на подходящ закон за “перфориране”, при което не се предават част от контролните битове.

Декодирането на турбо кодовете може да се осъществи с използването на стандартния алгоритъм на Витерби [3]. В този случай декодерът следва да определи най-правдоподобната предадена последователност от данни $\hat{\mathbf{u}}$ при известна приета последователност $\mathbf{y}$ на неговия вход, т.е.

$$\hat{\mathbf{u}} = \arg\{\max_{\mathbf{u}} P[\mathbf{u} \mid \mathbf{y}]\}. \quad (1)$$

В следствие на осъществяваната пермутация при турбо кодовете, сложността на алгоритъма на Витерби е $O(2^k)$, където k е дължината на пермутираните последователности от данни. Ето защо непосредственото прилагане на метода на максималното правдоподобие (т.е. използване на алгоритъма на Витерби) за декодиране на турбо кодове е неприложимо и следва да се използват субоптимални алгоритми с редуцирана сложност.

Основният алгоритъм за декодиране на турбо кодове използва посимволна оценка на данните по максимума на апостериорните вероятности [4]. В случай на турбо код с два компонентни конволюционни кодера този алгоритъм може да се опише със следната система уравнения:

$$L_i^1 = \log \frac{P[u_i = 1 \mid \mathbf{y}^0, \mathbf{y}^1, \mathbf{z}^2]}{P[u_i = 0 \mid \mathbf{y}^0, \mathbf{y}^1, \mathbf{z}^2]} \quad (2)$$

$$L_i^2 = \log \frac{P[\tilde{u}_i = 1 \mid \tilde{\mathbf{y}}^0, \mathbf{y}^2, \tilde{\mathbf{z}}^1]}{P[\tilde{u}_i = 0 \mid \tilde{\mathbf{y}}^0, \mathbf{y}^2, \tilde{\mathbf{z}}^1]}, \quad (3)$$

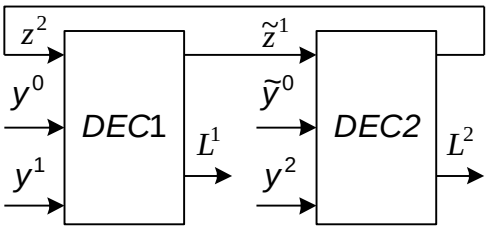
където $\mathbf{y}^0$ е приетата систематична последователност, $\mathbf{y}^1$ е приетата контрол-на последователност на първия кодер и $\mathbf{y}^2$ е приетата контролна последователност на втория кодер. Тилдите над променливите указват пермутирна версия на съответната променлива, т.е. $\tilde{\mathbf{y}}$ е пермутирана версия на $\mathbf{y}$. В (2) и (3) логаритмичните апостериорни отношения на правдоподобие на индексираните битове са означени с L_i^1 и L_i^2, а $\mathbf{z}^1$ и $\mathbf{z}^2$ дефинират коригиращата информация, реализирана в резултат от процеса на декодиране. Връзката на коригиращата информация с апостериорните отно-

шения на правдоподобие се дава с уравненията

(4) $$z_i^1 = L_i^1 - y_i^0 - z_i^2$$

(5) $$z_i^2 = \tilde{L}_i^2 - \tilde{y}_i^0 - \tilde{z}_i^1.$$

Системата уравнения (2) – (5) може да се реши итеративно с използване на структурата на турбо декодер, показана на Фиг.2.



Фиг.2. Структурна схема на турбо декодер.

На Фиг.2. *DEC*1 определя решението на (2), а *DEC*2 определя решението на (3). На всяка полуитерация единият декодер предава коригираща информация към другия декодер, който обновява апостериорните оценки на предадените битове (т.е. на всяка полуитерация коригиращата информация се използва като априорна от съответния декодер). Итеративното обновяване на апостериорните оценки на данните е възможно в следствие слабата корелация на коригиращата информация от двата декодера, като резултат от използваната пермутация в процеса на кодиране. Окончателна оценка на данните (“твърди” решения) се получава след определен брой итерации (като правило, след всяка итерация сходимостта на алгоритъма се подобрява) , при което

(6) $$u_i = \begin{cases} 1, \text{ако } L_i^2 > 0 \\ 0, \text{ако } L_i^2 < 0 \end{cases}.$$

Следва да се отбележи, че стандартният итеративен алгоритъм за посимволна оценка на данните по максимума на апостериорните вероятнсти [4] има значителна сложност и е чувствителен към крайната точност числовите резултати. Ето защо този алгоритъм се изпълнява в логаритмичната област [5], при което неговата сложност е приблизително два пъти по-голяма от тази на алгоритъма на Витерби.

3. Асимптотични характеристики на турбо кодовете.

Основен параметър за оценка на достоверността на предаваната информация при произволна цифрова комуникационна система е вероятността за грешка на бит P_b. Понеже разглежданите турбо кодове са линейни кодове, P_b може да се определи при допускането за предаване на изцяло нулева последователност от данни. Тогава вероятността за декодиране на *i*-тата кодова дума е

(7) $$P_b(i \mid \mathbf{0}) = \frac{w_i \, Q\left(\sqrt{2Rd_i E_b / N_0}\right)}{k},$$

където w_i е информационното Хемингово тегло (броят на информационните единици) в *i*-тата кодова дума, *Q* е Гаусовата *Q*-функция, *R* е скоростта на турбо кода, d_i е Хеминговото тегло на *i*-тата кодова дума, E_b / N_0 е отношението сигнал/шум на информационен бит, а *k* е дължината на информационния блок. Вероятността за грешка на бит може да се ограничи отгоре [6], при което

(8) $$P_b \le \sum_{i=1}^{2^k} \frac{w_i \, Q\left(\sqrt{2Rd_i E_b / N_0}\right)}{k}.$$

За средни и високи отношения сигнал/шум сумата в (8) се доминира от членовете, в който участва минималното Хемингово разстояние на турбо

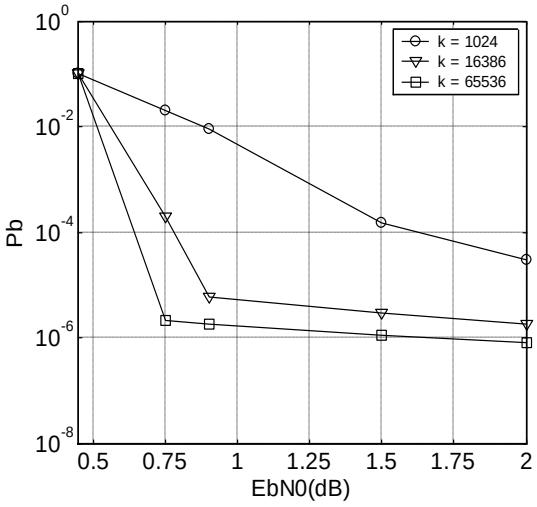
кода $d_{\min}$. Тогава асимптотичното поведение на турбо кодовете може да се оцени, като

$$P_b \approx \frac{n_{\min} w_{\min} Q\left(\sqrt{2Rd_{\min} E_b / N_0}\right)}{k}, \quad (9)$$

където $n_{\min}$ е броят на кодовите думи с Хемингово тегло $d_{\min}$, а $w_{\min}$ е средното Хемингово тегло на информационните последователности, които пораждат кодови думи с минимално Хемингово тегло. При използване на псевдослучайни закони за пермутация параметрите на турбо кода $d_{\min}$ и $n_{\min}$ се определят от информационни последователности с Хемингово тегло две (т.е. $w_{\min} = 2$) [7]. Тогава асимптотичното поведение на турбо кодовете може да се определи с (9), като се намерят всички кодови думи с минимално Хемингово тегло, които се генерират от информационни последователности с Хемингово тегло две.

4. Характеристики и приложения на турбо кодовете.

Характеристиките и приложенията на турбо кодовете се определят от значителен брой фактори, като основните от тях са: дължината на пермутираната информационна последователност; скоростта на турбо кода; методът за декодиране; внасяното закъснение в процеса кодиране /декодиране. Вероятностните характеристики на турбо кодовете се определят на първо място от дължината *k* на пермутираната информационна последователност. На Фиг.3. е показана вероятността за грешка на бит P_b на оригиналния турбо код [1] с генераторни полиноми $g_0 = (37)_8$, $g_1 = (21)_8$ и скорост 1/2 за дължина на информационната последователност 1024, 16384 и 65536 бита.



Фиг.3. Вероятност за грешка на бит на турбо код с генераторни полиноми $g_0 = (37)_8$, $g_1 = (21)_8$ и скорост *R* = 1/2.

Съгласно Фиг.3., с увеличаването на *к* се подобрява вероятността за грешка на бит P_b на турбо кода при ниски отношения сигнал/шум. Това се обяснява с факта, че при ниски отношения сигнал/шум коефициента $(n_{\min} w_{\min} / k) << 1$ в (9) доминира вероятностната характеристика на турбо кода. От друга страна, за средни и високи отношения сигнал/шум се реализира "праг" на P_b в следствие на относително малките стойности на минималното Хемингово разстояние $d_{\min}$ на турбо кода (за разглежданият турбо код $d_{\min} = 6$ при *k* =65536 бита).

Типичните стойности на скоростта *R* на турбо кодовете са в диапазона от 1/4 до 1/2. В посоченият диапазон от скорости характеристиките на турбо кодовете се доближават на десети от децибела до теоретичните граници на Шенон за комуникационен канал с адитивен бял Гаусов шум (при относително големи стойности на *k*).

Както беше отбелязано, основният алгоритъм за декодиране на турбо кодовете се заключава в итеративно обновяване на "меките" решения за данните с използване на апостериорните вероятности. Към настоящият

момент са предложени значителен брой субоптимални алгоритми за "меко" итеративно декодиране на турбо кодове с редуцирана сложност [5], като енергийната загуба най-често е в границите от 0,2 dB до 1,5 dB.

Параметърът k влияе непосредствено на възможните комуникационни приложения на турбо кодовете във връзка с внасяното закъснение при декодирането. Това закъснение се определя приблизително с отношението k/R_b, където R_b [бит/секунда] е информационната скорост на източника. Така например, при типична скорост на речевия кодер в мобилните комуникационни средства от 8000 бита за секунда и допустимо закъснение от 40 милисекунди, дължината на пермутираната информационна последователност не трябва да надвишава 320 бита.

Към настоящият момент съществуват значителен брой рекомендации за кодиране на телеметрични, спътникови и мобилни комуникационни канали с използване на турбо кодове [8], [9], [10], а други са в процес на разработване. Използването на турбо кодове позволява да се реализира енергийна печалба от 2-3 dB и повече спрямо съществуващите комуникационните системи с "конвенционални" коригиращи кодове при запазване на високо качество на предаваната информация.

5. Заключение.

Турбо кодовете са най-новото достижение в областта на коригиращите кодове. Характеристиките на тези кодове се доближават до теоретичните граници на Шенон при сложност на декодирането, която позволява практическо им използване. Оптимизирането на турбо кодовете за специфични комуникационни приложения с прилагане на ефективни софтуерни и хардуерни решения за декодиране е ключът към тяхното още по широко практическо приложение.

Литература

[1] C. Berrou, et al, "Near Shannon limit error-correcting coding and decoding: turbo-codes," *ICC* 1993, Geneva, Switzerland, pp.1064-1070, May 1993.

[2] C. Shannon, "Probability of error for optimal codes in a Gaussian channel," *Bell Syst. Tech. J.*, 38, pp.611–656, May 1959.

[3] G. Forney, "The Viterbi Algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268-278, March 1973.

[4] L. Bahl, et al, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284-287, March 1974.

[5] P. Robertson, et al, "Optimal and Sub-Optimal Maximum a Posteriori Algorithms Suitable for Turbo Decoding," *European Trans. Telecommun*., vol. 8, no. 2, pp.119-125, March-April 1997.

[6] S. Lin and D. Costello, "Error control coding: fundamentals and applications," Prentice-Hall, 1983.

[7] S. Benedetto and G. Montorsi, "Design of Parallel Concatenated Convolutional Codes," *IEEE Trans. Commun.*, vol.44, no.5, pp.591-600, May 1996.

[8] CCSDS, "Telemetry Channel Coding," Blue Book, vol. 101.0-B-4, issue 4, May 1999.

[9] ETSI PR EN 301 790, "Digital video broadcasting (DVB): Interaction channel for satellite distribution systems," V1.1.1, April 2000.

[10] ETSI TS 125 212, "Universal Mobile Telecommunications Systems (UMTS); Multiplexing and Channel Coding (FDD)," V3.2.0, March 2000.

Компютърно-математически метод за синтез на оптимални сингулярни адаптивни компютърни наблюдатели – част I

Любомир Н. Сотиров, Стела Л. Сотирова

Резюме: Предложен е компютърно-математически метод за алгоритмичен синтез на новото поколение адаптивни компютърни наблюдатели, включващи оптимални оценители на вектора на началното състояние, идентификатори на параметрите и оптимални сингулярни (пълни и дегенеративен) адаптивни компютърни наблюдатели на вектора на текущото състояние за линейни стационарни дискретни системи с един вход и един изход. Оценяването на елементите на вектора на началното състояние не е присъщо на методите за синтез на адаптивни наблюдатели, базирани само на идентификатори. Важно е отпадането на процедурата за синтез на системи със зададени полюси, затрудняваща компютърното приложение на резултатите от теорията на наблюдението на Луенбергер и методите за синтез на адаптивни наблюдатели, базирани само на идентификатори. Синтезираният оптимален сингулярен адаптивен компютърен наблюдател има свойства като едрозърнест и естествен паралелизъм, които позволяват неговата софтуерна и хардуерна интерпретация също и с паралелно-векторни компютърни архитектури. Конструираният алгоритъм е интерпретиран и в MATLAB-среда.

A Computer – Mathematical Method for Synthesis of Optimal Singular Adaptive Computer Observers – Part I

Lyubomir N. Sotirov, Stela L. Sotirova

Abstract: A computer- mathematical method for algorithmic synthesis of the new generation of adaptive computer observers, including optimal estimators of the initial state vector, identifiers of the parameters and optimal singular (full and degenerate) adaptive computer observers of the current state vector for single-input single-output linear stationary discrete systems is suggested. The estimation of the initial state vector elements is not typical of the methods for synthesis of adaptive observers, based only on identifiers. The important fact is the dropping off of the procedure for the synthesis of systems with given poles, hindering the computer application of the results from Luenberger's observation theory and the methods for synthesis of adaptive observers, based only on identifiers. The synthesized optimal singular adaptive computer observer has features such as large- scale granularity and natural parallelism, which also allow its software and hardware implementation on a parallel- vector computer architecture. The constructed algorithm is implemented in MATLAB- environment.

1. Постановка на задачата за оптимално сингулярно адаптивно наблюдение

Разглеждат се наблюдаеми линейни дискретни системи от вида:

$$(1) \qquad \mathbf{x}(k+1)=\mathbf{A}\mathbf{x}(k)+\mathbf{b}u(k),\quad \mathbf{x}(0)=\mathbf{x}_0$$

$$(2) \qquad y(k)=\mathbf{c}^T\mathbf{x}(k),\quad k=0,1,2,\ldots,$$

където $\mathbf{x}(k)\in R^n$ е неизвестен вектор на текущото състояние, $\mathbf{x}(0)\in R^n$ е неизвестен вектор на началното състояние, $u(k)\in R^1$ е скаларен вход, $y(k)\in R^1$ е скаларен изход, **A** е неизвестна матрица от вида:

(3) $$\mathbf{A}=\begin{bmatrix} \mathbf{0} & \vdots & \mathbf{I_{n-1}} \\ \cdots & \cdots & \cdots \\ & \mathbf{a}^{\mathrm{T}} & \end{bmatrix},$$

$\mathbf{a}^{\mathrm{T}}=[a_1,\ a_2, \ldots, a_n]$,

където $\mathbf{I_{n-1}}$ е единична $(n-1)$x$(n-1)$ матрица, **b** е неизвестен вектор от вида:

(4) $\mathbf{b}^{\mathrm{T}}=[b_1,\ b_2, \ldots, b_n]$,

c е вектор от вида:

(5) $\mathbf{c}^{\mathrm{T}}=[1, 0, \ldots, 0]$.

На уравненията *(1)*, *(2)*, интерпретиращи поведението на разглежданите линейни стационарни дискретни системи в пространството на състоянията, съответства следното диференчно уравнение "вход-изход:

(6) $$\begin{aligned} &y(k+n)-a_n y(k+n-1) - a_{n-1}y(k+n-2) - \cdots - a_2 y(k+1) - a_1 y(k)= \\ &=h_1 u(k+n-1)+ h_2 u(k+n-2)+ \cdots + h_{n-1}u(k+1)+h_n u(k), \\ &k=0,1,2,\ldots, \\ &y(0)=y_0,\ y(1)=y_1, \cdots, y(n-1)=y_{n-1}, \end{aligned}$$

както и следната предавателна функция:

(7) $$\mathbf{W}(z)=\frac{\mathbf{Y}(z)}{\mathbf{U}(z)}=\frac{h_1 z^{n-1}+h_2 z^{n-2}+\cdots+h_{n-1}z+h_n}{z^n-a_n z^{n-1}-\cdots-a_2 z-a_1}.$$

Задачата за оптимално сингулярно адаптивно (ОСА) наблюдение се състои в алгоритмичния синтез на новото поколение адаптивни компютърни наблюдатели, включващи идентификатори на неизвестните векторни параметри **a**, **h** и **b,** оптимални оценители на неизвестния вектор на началното състояние **x***(0)* и ОСА наблюдатели (пълни и дегенеративен) на неизвестния вектор на текущото състояние **x***(k)*, *k=1,2,...*

Подобни постановки на задачата за адаптивно наблюдение са формулирани в непрекъснатия случай от Carroll и Lindorff (1973), Kudva и Narendra (1973), Lüders и Narendra (1973, 1974), Nuyan и Carroll (1976), Kreisselmeier (1977), Nikiforuk и др. (1977).

Narendra и Kudva (1974) проектират дискретен адаптивен наблюдател, използвайки директния метод на Ляпунов. Suzuki и Andoh (1977) също разработват дискретен адаптивен наблюдател, въз основа на теоремата за хиперустойчивост на Попов. Shahrokhi и Morari (1982) предлагат дискретен адаптивен наблюдател, базиран на идентификатор с произволно висока скорост на сходимост.

Murdoch и Oliveros (1985) синтезират дискретен адаптивен наблюдател за система от неизвестен ред. Дискретен адаптивен наблюдател с експоненциални тегловни характеристики разработват Hang, Kim и Choi (1989).

Тези проблеми са решени по съответния оригинален начин, но авторите не дават отговор на въпроса как може да се оцени неизвестния начален вектор на състоянието (и в последствие това да се използва за оптимизация и на оценката на неизвестния текущ вектор на състоянието).

Неизвестният текущ вектор на състоянието може да бъде оценен и с идентичния наблюдател на Luenberger (1971), чиито качества могат да бъдат подобрени, ако векторните параметри на наблюдавания обект и неговото начално състояние се оценяват, например, с инструментариума на теорията за ОСА наблюдение.

За решаване на формулираната задача за адаптивно наблюдение в стохастическия случай може да се използва и разширения естиматор на Kalman, но той също не оценява неизвестния начален вектор на

състоянието (с всички, произтичащи от тук последици).

Известно е (Arimoto и Hino, 1974), че при синтеза на оптимални системи, въз основа на квадратични функционали, показателят на качеството на системата се влошава, ако се използва конвенционален наблюдател на текущото състояние, при неточно познаване на началния вектор на състоянието на оптимизируемата система. При това се допуска, че векторните параметри на обекта са точно известни, което на практика не винаги се изпълнява.

Интересно е да се отбележи, че например от Eykhoff (1974) се твърди, че задачата за съвместна идентификация и наблюдение "неизбежно има нелинеен характер", докато приведеното по-нататък решение говори, че в редица случаи (достатъчно общи) това твърдение не е в сила.

Синтезът на адаптивни компютърни наблюдатели по-долу се извършва с помощта на инструментариума на теорията за оптимално сингулярно адаптивно наблюдение.

Впечатление за състоянието и развитието на теорията за оптимално сингулярно адаптивно наблюдение може да се получи от Л. Сотиров /1980-2002/, от Л.Сотиров и Г.Сухов / 1997 / и от Л.Сотиров и С.Сотирова / 2002 /.

2. M1A6-алгоритъм за едновходови линейни стационарни дискретни системи с оценяване на началния вектор на състоянието

Стъпка 1. Формират се вектори и матрици от входно-изходни данни:

$Y_1^T=[y(0),y(1),\ldots,y(n-1)]$,

$Y_2^T=[y(n),y(n+1),\ldots,y(2n-1)]$,

$Y_3^T=[y(2n),y(2n+1),\ldots,y(3n-1)]$,

$$\overset{\Delta}{U}=\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ u(0) & 0 & \cdots & 0 & 0 \\ u(1) & u(0) & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ u(n-2) & u(n-3) & \cdots & u(0) & 0 \end{bmatrix},$$

където $\overset{\Delta}{\mathbf{U}}$ е ляво-триъгълна *n*x*n* матрица на Теплиц,

$$\mathbf{U}_{11}=\begin{bmatrix} u(n-1) & u(n-2) & \cdots & u(1) & u(0) \\ u(n) & u(n-1) & \cdots & u(2) & u(1) \\ u(n+1) & u(n) & \cdots & u(3) & u(2) \\ \vdots & \vdots & & \vdots & \vdots \\ u(2n-2) & u(2n-3) & \cdots & u(n) & u(n-1) \end{bmatrix},$$

$$\mathbf{U}_{21}=\begin{bmatrix} u(2n-1) & u(2n-2) & \cdots & u(n+1) & u(n) \\ u(2n) & u(2n-1) & \cdots & u(n+2) & u(n+1) \\ u(2n+1) & u(2n) & \cdots & u(n+3) & u(n+2) \\ \vdots & \vdots & & \vdots & \vdots \\ u(3n-2) & u(3n-3) & \cdots & u(2n) & u(2n-1) \end{bmatrix},$$

където $\mathbf{U}_{11}$ и $\mathbf{U}_{21}$ са *n* x *n* матрици на Теплиц,

$$\mathbf{Y}_{12}=\begin{bmatrix} y(0) & y(1) & \cdots & y(n-2) & y(n-1) \\ y(1) & y(2) & \cdots & y(n-1) & y(n) \\ y(2) & y(3) & \cdots & y(n) & y(n+1) \\ \vdots & \vdots & & \vdots & \vdots \\ y(n-1) & y(n) & \cdots & y(2n-3) & y(2n-2) \end{bmatrix},$$

$$\mathbf{Y}_{22}=\begin{bmatrix} y(n) & y(n+1) & \cdots & y(2n-2) & y(2n-1) \\ y(n+1) & y(n+2) & \cdots & y(2n-1) & y(2n) \\ y(n+2) & y(n+3) & \cdots & y(2n) & y(2n+1) \\ \vdots & \vdots & & \vdots & \vdots \\ y(2n-1) & y(2n) & \cdots & y(3n-3) & y(3n-2) \end{bmatrix},$$

където $\mathbf{Y}_{12}$, $\mathbf{Y}_{22}$, са n x n - матрици на Ханкел.

Стъпка 2. Оценките на векторите **h** и **a** се изчисляват, използвайки линейната система алгебрични уравнения от вида:

$$\Omega\hat{\Theta}=Y,$$

където:

$$\hat{\Theta}^T=\left[\hat{h}^T:\hat{a}^T\right],$$

$$\mathrm{Y}^{\mathrm{T}}=\left[\mathrm{Y}_2^{\mathrm{T}}:\mathrm{Y}_3^{\mathrm{T}}\right],$$

$$\Omega=\begin{bmatrix} U_{11} & \vdots & Y_{12} \\ \cdots & \cdots & \cdots \\ U_{21} & \vdots & Y_{22} \end{bmatrix}$$

Стъпка 3. Оценката на вектора **b** се изчислява, използвайки следната линейна система от алгебрични уравнения:

$$T\hat{b}=\hat{h},$$

където:

$$\mathbf{T}=\begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -\hat{a}_n & 1 & \cdots & 0 & 0 \\ -\hat{a}_{n-1} & -\hat{a}_n & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ -\hat{a}_2 & -\hat{a}_3 & \cdots & -\hat{a}_n & 1 \end{bmatrix}$$

Стъпка 4. Началният вектор на състоянието се изчислява с помощта на оптималния оценител от вида:

$$\hat{\mathrm{x}}(0)=\mathrm{Y}_1-\overset{\Delta}{\mathrm{U}}.\hat{\mathrm{b}}$$

Стъпка 5. Текущият вектор на състоянието се оценява, използвайки следния пълен оптимален сингулярен адаптивен наблюдател от вида:

$$\hat{x}(k+1)=\hat{A}\hat{x}(k)+\hat{b}u(k)+g\left[y(k)-c^T\hat{x}(k)\right],$$

$$\hat{x}(0)=\hat{x}_{0,}$$

$k=0, 1, 2, ...,$

където:

$$\mathbf{g}^{\mathrm{T}}=[g_1, g_2, ..., g_n]$$

или от вида:

$$\hat{\mathbf{x}}(k+1)=\hat{\mathbf{F}}\hat{\mathbf{x}}(k)+\hat{\mathbf{b}}u(k)+\mathbf{g}y(k),$$

$$\hat{\mathbf{x}}(0)=\hat{\mathbf{x}}_0,$$

$k=0, 1, 2, ...,$

където:

$$\hat{\mathbf{F}}=\hat{\mathbf{A}}-\mathbf{g}\mathbf{c}^{\mathrm{T}}.$$

Изборът на елементите на вектора **g** е произволен и може да бъде осъществен, например, по един от следните начини:

1) $\mathbf{g}=\hat{\mathbf{b}}$,

2) **g** се избира така, че матрицата $\hat{\mathbf{F}}$ да има собствени стойности, разположени в единичния кръг,

3) **g=0,**

4) $\mathbf{g} = \hat{\mathbf{h}}$.

Стъпка 6. Текущият вектор на състоянието може също да бъде оценен като се използва дегенеративния (при g=0) ОСА наблюдател от вида:

$$\hat{\mathbf{x}}(k+1) = \hat{\mathbf{A}}\hat{\mathbf{x}}(k) + \hat{\mathbf{b}}u(k), \quad \hat{\mathbf{x}}(0) = \hat{\mathbf{x}}_0,$$
$$k=0, 1, 2, ...,$$

който може да се разглежда като изроден случай на пълния ОСА наблюдател и може да се конструира по-лесно.

От синтезирания алгоритъм непосредствено следва:

Теорема M1A6. Единствено решение на задачата за ОСА наблюдение, с помощта на алгоритъм **M1A6**, съществува, ако и само ако матрицата Ω е неособена или

$$\det \Omega \neq 0.$$

ЛИТЕРАТУРА

1. Sotirov, L. N., 1994, A direct method for adaptive observation of single-input single- output linear stationary discrete systems with initial state vector estimation, Reports of the Bulgarian Academy of Sciences, vol. 47, No. 11, 5-8, Sofia.

2. Sotirov, L. N., 1997, Optimal singular adaptive observation of stationary discrete systems with initial state vector estimation. *Automation and Remote Control, Russian Academy of Sciences, No 9, 110-118, Moscow, New York, London .*

3. Sotirov, L. N., 1997, An algorithm for synthesis of optimal singular adaptive observers with direct estimation of the initial state vector. *International Journal of Systems Science, vol. 28, No 6, 559-562.*

4. Sotirov, L. N., E. G. Sukhov, 1997, A parallel direct method for optimal adaptive estimation of discrete linear system, *Automation and Remote Control, Russian Academy of Sciences, No 10, 154-163, Moscow, New York, London .*

5. Sotirov L.N., 1999, Reduced order optimal singular adaptive observation for a class of discrete systems, *Automation and Remote Control, Russian Academy of Sciences, No.2, 75-82, Moscow, New York, London*.

6. Sotirov L.N., 1999, An algorithm for synthesis of discrete reduced order optimal singular adaptive observers. *International Journal of Systems Science* , vol.30, No. 6, 665-671.

7. Sotirov L.N., 2000, Optimal singular adaptive observation of multi- input discrete linear systems with perturbation, *Automation and Remote Control, Russian Academy of Sciences, No 7, 120-130, Moscow, New York, London .*

8. Sotirov L.N., 2001, An algorithmic synthesis of discrete optimal singular adaptive observers for multiple- input single- output linear systems, . *International Journal of Systems Science* , vol.12, No. 5, 545-551.

9. Sotirov L.N., 2001, An algorithm for synthesis of discrete time- optimal adaptive controllers, Reports of the Bulgarian Academy of Sciences, vol. 54, No. 3, 19-24, Sofia.

10. Sotirov L.N., S.L. Sotirova, 2002, An algorithmic synthesis of discrete optimal singular adaptive observers for single- input single- output linear systems, Reports of the Bulgarian Academy of Sciences, vol. 55, No. 7, 31-36, Sofia.

11. Сотиров Л. Н., Избрани глави от съвременната теория на управлението, 1998, Технически университет, Варна.

12. Сотиров Л. Н., Теория на автоматичното управление, Част 2 / Теория на дискретните системи за автоматично управление /, 2000, Технически университет, Варна.

Възстановяване на изображения по пространствено подобие

Марияна Цв. Стоева

Резюме: В доклада се предлага метод за възстановяване на изображения от БДИ по пространствено подобие, базиран на геометрична структура, даваща възможност за извличане на пространствени релации между областни обекти, непроменливи по отношение трансформациите транслация, ротация, мащабиране, рефлексия (симетрия), смяна на гледната точка, както и произволни композиции от тези преобразувания.

Retrieval of Images by Spatial Similarity

Mariana Cv. Stoeva

Abstract: This paper presents our work in the area of image retrieval in Image Databases for images saved by spatial similarity of domain objects location. We propose a geometry-based structure, which makes possible the extraction of directional spatial relations among domain objects that are invariant to transformations such as translation, rotation, scaling, reflection, as well as arbitrary compositions of these transformations.

1. Увод

Докладът представя работа ни в областта на възстановяването на изображения, съхранени в База Данни от Изображения (БДИ) по пространствено подобие на разположението на областните обекти, които съдържат. Предлагаме метод за възстановяване на изображения от БДИ, базиран на геометрична структура, даваща възможност за извличане на пространствени релации между областни обекти, непроменливи по отношение трансформациите транслация, ротация, мащабиране, рефлексия, смяна на гледната точка, както и произволни композиции от тези преобразувания.

Разширените обекти на всяко изображение са съхранени в БДИ със символни имена и координати на 5 характерни за формата на обекта точки. Тази съхранена за всеки обект информация дава възможност да се апроксимира формата на разширените обекти със сектори от пръстени, определени от концентрични окръжности. Тази апроксимация дава възможност да се определят аналогично на ортогоналните модели широко известните 13 пространствени релации в една линейна посока и една специфично използвана кръгова посока. Осигурява се непроменливост относно трансформации на определяне на атомичнитепространствени релации между два обекта, като се използват свойствата на Центъра на Тежестта (ЦТ) на обекта и ЦТ на изображението.

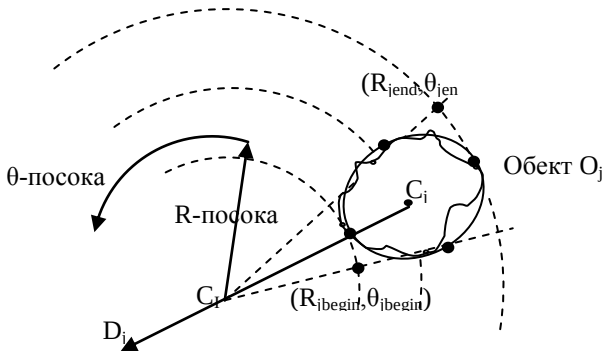
Въвежда се една подобностна мярка разпознаваща трансформирани изображения и под изображения. Временната сложност на алгоритъма, реализиращ възстановяването на изображения от БДИ е n^2 , където n е броя на обектите, общи за заявковите и БДИ изображения. Обработката е стабилна в смисъл, че може да разпознае транслационни, мащабирани ротационни и рефлексни вариантни изображения и вариантите, създадени от произволно композиране на тези гео-

метрични трансформации. Ефикасността на подобностно възстановяване по пространствено подобие на областни обекти са оценени експериментално.

2. Описание на изображението и използвани апроксимации

От информацията в едно изображение представена на пикселно ниво след прилагане на различни сегментиращи обработки и разбиращи техники се идентифицират областните обекти и тяхната локация в изображението. Чрез обединяване на едно име с всеки от областните обекти се получава едно символо изображение. При пространствените изследвания най-често се използва приближение на формата на обектите наречено Минимално Обхващащ Правоъгълник (МОП - най-малкия правоъгълник обхващащ обекта). Използването на МОП за представянето на обектната формата не позволява получаването на пространствени релации, непроменливи по отношение на ротация и рефлексия. В нашата работа търсим такова приближено кратко представяне на формата на обектите, което да бъде непроменливо по отношение на произволни композиции от преобразувания и в същото време да запазва информацията, необходима за отчитане на пространствените релации между разширените обекти в едно изображение. Това приближено представяне се получава от 5 съхранени характерни за обекта точки, чието определяне е непроменливо по отношение на трансформации и в същото време представят формата на обекта.

Приемаме, че едно изображение I се състои от n областни обекта, означени като O_j. В БДИ се съхранява символното описание на изображение I. Нека C_J е ЦТ на обекта O_j, а C_I е ЦТ на изображението получен от ЦТ на съдържащите се в изображението обекти. Описанието представя едно изображение като $I=((O_j\ (X_{cj},Y_{cj}),\ (O_j.x_1,\ O_j.y_1),\ (O_j.x_2,\ O_j.y_2),\ (O_j.x_3,\ O_j.y_3),\ (O_j.x_4,\ O_j.y_4)),\ 1\le j\le n)$, където: O_j е наименование на обекта, (X_{cj},Y_{cj}) са координатите на ЦТ C_j на обект O_j, а $(O_j.x_1,\ O_j.y_1,\ O_j.x_2,\ O_j.y_2,\ O_j.x_3,\ O_j.y_3,\ O_j.x_4,\ O_j.y_4)$ са координати на характерните за формата на обекта координати на 4 точки от външния контур, описващ обекта, по отношение на картезианската координатна система с начало C_j. Характерните точки са пресечните точки на главната и допълнителна ос на обекта с минимално обхващащия обекта елипсоид, апроксимиращ формата на обекта. Приемаме че областните обекти са съхранени по азбучен ред на имената на обектите. Приемаме, че областните обекти са именувани последователно през всички изображения в БД и изображенията съдържат многократни примери от обектен тип.



Фиг.1 Илюстация на използваната апроксимация

Така областните обекти на всяко изображение са съхранени в БДИ в символен вид и координати на точки, чието определяне е непроменливо по отношение на трансформации [2] и в същото време характеризират формата на обекта.

Апроксимацията на формата на разширения обект O_j построена по съхранената за всеки обект информация има вида на сектор от пръстен получен от концентрични окръжности около ЦТ на изображението C_I с

граници определени от минимумите (r_{jbegin}, θ_{jbegin}) и максимумите, (r_{jend}, θ_{jend}) на полярните координати спрямо този център на точките от четириъгълника определен от координатите на съхранените 4 характерни за обекта точки. Началната точка на сектора от пръстен има полярни координати (R_{begin}, θ_{begin}), а крайната полярни координати (R_{end}, θ_{end}). На Фиг.1 е показан пример на сектор от пръстен апроксимиращ формата на обект А, описан със своите 4 характерни точки. За получаването на пространствените релации на всеки обект с останалите обекти от изображението се използват полярните координати на ЦТ на обектите и полярните координати на началната и крайна точка на апроксимиращите ги сектори от пръстени.

3. Пространствено подобностно възстановяване на изображения от БДИ

Следвайки философията на заявка чрез изобразителен пример, заявката към БДИ се обработва чрез двойковане на извлечените от нейното описание пространствени релации срещу тези на изображенията съхранени в БДИ. Изследвания, отнасящи се до методологиите за подобностно възстановяване могат да се намерят в [4], [5], [6].

За определяне на пространствените връзки между всяка двойка областни обекти в едно изображение използваме дирекционен метод. Из-ползваме две посоки, съответстващи на полярна координатна система с център ЦТ на изображението. Първа-та посока има линейно сканиращо направление (R-посока), започващо от ЦТ на изображението C_I и съответстващо на концентрични окръжности, движещи се началото навън. Втората кръгова посока (θ-посока) съответства на трасе извиващо се по посока обратна на часовниковата стрелка. Посоките са представени на Фиг. 1. При определяне на връзките между всяка двойка обекти в смисъл на условия върху началните и крайните им точки във всяка отделна посока се определят широко известните 13 прос-транствени ортогонални релации [1], чието дефиниране е адаптирано към полярна координатна система. В двете посоки, възникват общо 169 пространствени връзки между два обекта в две димeнсии, чрез които пространственото съдържание може да се представи удобно. Пространствените релации между обектите в изображението се изчисляват като се използват полярните координати на началната (R_{begin}, θ_{begin}) и крайна точка (R_{end}, θ_{end}) на апроксимиращите обектите сегменти от пръстени с център ЦТ на изображението.

Определение1. Една подредена тройка (O_j γ O_i) се нарича атомична пространствена релация, където обектите O_j, $O_i \in O_I$ и оператора **γ∈{<,>,|,:,=,[,(,],),/,\,%,#}**. Използва означението ($O_j \gamma_R O_i$) за да се покаже, че двойката обекти (O_j,O_i) принадлежи на релация γ в R-посока и означението ($O_j \gamma_\theta O_i$) да се покаже, че двойката обекти (O_j, O_i) принадлежи на релация γ в θ-посока.

Използвани са 7-те символа за пространствени оператори използвани от Ли и Хсю [5] и са въведени 4 нови знака. Така преизчислени начални и крайни точки на сектора от пръстена описващ областния обект релациите остават непроменливи по отношение на трансформацията ротация. При трансформация рефлексия релациите могат лесно да се получат, тъй като отношенията “по-голямо” и “по-малко” се променят алтернативно.

Тъй като целта е да се възстановят онези изображения, които съдържат почти същите обекти с подобно пространствено подреждане нашата подобностна мярка трябва да отчита подо-

бията между пространствените релации.

Определение 2. Нека двойката обекти (O_j,O_i) в изображение Q принадлежи на пространствената релация γ (O_j γ O_i) и двойката обекти (O_j,O_i) в изображение I принадлежи на пространствената релация γ' (O_j γ' O_i). Подобието между пространствените релации в двете изображения Q и I за двойката обекти (O_j,O_i) определяме като $sim_{ji}(\gamma,\gamma')=t$, където $t\in[0,1]$ е подобието между пространствените оператори γ и γ'. Означаваме подобието между релациите в двете изображения Q и I за двойката обекти (O_j,O_i) в R-посока като $sim_{ji}(\gamma_R,\gamma_R')$ и като $sim_{ji}(\gamma_\theta,\gamma_\theta')$ в θ-посока.

Приспособяваме интервалния съседствен граф от [5], [3], който дефинира дистанциите между пространстве-ните оператори, съгласно правилото, дадено от Фреска в [3], две проекционни връзки да са съседи ако могат да се трансформират една в друга чрез непрекъснато деформиране на проекциите. В [7] са представени стойностите на подобие $\{sim_{ji}(\gamma,\gamma')\}$, между всяка двойка оператори, които използваме.

Двойковането изисква дефиниране на ефикасна подобностна мярка, която е ефикасна за изчисляване и улавяне на задължителните аспекти на подобие, които хората използват . За тази цел, въвеждаме една формула, която измерва степента на подобие между изображение от БДИ и заявка в смисъл на обекти и пространствени връзки между тях и я изразява като стойност. Използваната от нас подобностна мярка отразява нашето разбиране за подобие и включва обектен и пространствен фактор. Обектният фактор отчита наличието на общи и за двете изображения обекти. Пространственият фактор отчита подобието на пространственото разположение на общите и за двете изображения обекти, чрез подобието съответстващите им атомични релации.

Определение 3. Нека зявковото изображение е Q, а изображението от БДИ е I. Подобностната дистанция между Q и I дефинираме с уравнение (1).

$$sim(Q,I)=\frac{1}{m}\left(n+\frac{1}{2n}\sum_{j}^{n}\sum_{i}^{n}sim_{ji}(\gamma_R,\gamma_R')+sim_{ji}(\gamma_\theta,\gamma_\theta')\right) \quad (1)$$

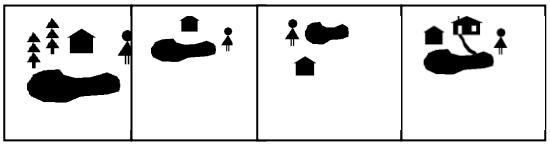
където: $m=\|O_Q\|$ е броя на обектите в заявковото изображение, $n=\|O_Q \cap O_I\|$ е броя на общите и за двете изображения обекти, $sim_{ji}(\gamma_R,\gamma_R')$ е пространственото подобие между двете изображения Q и I за двойката обекти (O_j,O_i) в R-посока, $sim_{ji}(\gamma_\theta,\gamma_\theta')$ е пространственото подобие между двете изображения I Q за двойката обекти (O_j,O_i) в θ-посока.

Едно изображение I от БД удовлетворява заявката Q само ако съдържа всички обекти от заявката и със същите релации като в изображение Q. Колкото повече обекти от Q се съдържат в изображението I, толкова по-висока е степента на подобие. Тази формула връща стойност от порядъка на [0,2] чрез изчисляване на съотношението между количеството обекти, възстановени заедно със стойността на подобие на техните атомични релации и количеството обекти изисквани от потребителя и техните атомични релации. Очакваме стойност 2 когато двойкованите изображения са идентични, в противен случай функцията ще приеме по ниски стойности. Тези стойности са пропорционални на степента на несъответствие в пространствените връзки между съответните обекти във входните изображения.

4. Експерименти и резултати

За целите на експеримента използваме тестова изобразителна колекция от 4 оригинални изображения и 4 ва-

рианта на всяко от тези изображения. Вариантите са генерирани с трансформационни оператори приложени върху всички областни обекти, както и върху подгрупа от областни обекти.

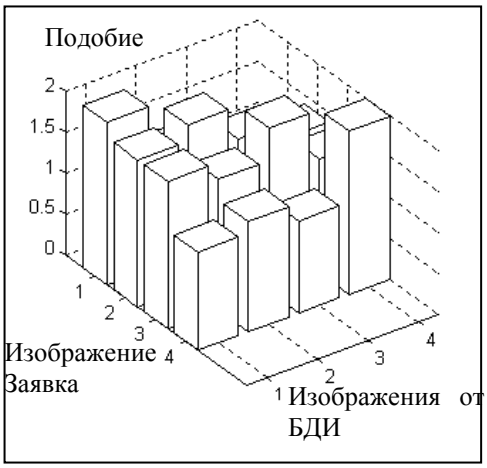


Фиг.2. Тестови изображения.

Експерименталната оценка се фокусира върху пространственото подобие изчислено чрез предлагания метод и стабилността на обработката по отношение на смесени трансформации. За да се установи дали обработката на предлагания метод получава като резултат подредени по пространствено подобие изображенията като отговор на едно заявково изображение по начин, който съответства на нашето интуитивно визуално подреждане, разглеждаме изображенията показани на Фиг.2 Използваме всяко изображение като заявка и изчисляваме неговото подобие с всички останали изображения. Получената оценка за пространствено подобие е представена с хистограмите на Фиг.3 и съответства на интуитивно очакваните стойности. Експериментът показа, че обработката на изобразителната заявка изпълнява очакванията ни за стойности на подобностната мярка.

5. Заключение

Предлаганият дирекционен метод за съхраняване и възстановяване на изображения от БДИ по пространствено подреждане на обектите, които съдържат, притежава инвариантност относно произволни трансформации, способност силно да улавя пространственото подобие, висока ефективност на съхраняването на пространствената информация и добра ефективност на обработката на пространствените заявки.



Фиг.3. Пространствено подобие на двойкованитете изображениея от Фиг.3.

Литература

1. J.F.Allen, Maintaining Knowledge about temporal Intervals: Comm. ACM, vol. 26, 832-843, 1983

2. C.Faloutsos, M.Ranganathan, Y. Manopolous, Fast Subsequence Matching in Time-Series Database, Proc ACM SIGMOD Int'l Conf. Managment of data, Mineapolis, Minn, 1999

3. C.Freksa, Temporal Reasoning Based on Semi-Intervals, Artificial Intelligence, vol. 54(1-2) 199-227, 1992

4. V.Gudivada, θℜ-stribg A geometry-based representation for efficient and Effective Retrieval of Images by Spatial Similarity, IEEE TKDE, vol.10, no.3, 504-511, 1998

5. S.Y.Lee, F. J. Hsu, Spatial Reasoning and Similarity Retrieval of image Using 2D C-string Knowledge Representation, Pattern Recognition, 25(10), 305-318, 1992

6. G.Petraglia, M. Sebillo, M.Tucci, G.Tortora, Virtual Images for Similarity Retrieval in Image Databases, IEEE TKDE vol. 13, no. 6, pp. 951-967,2001

7. M.Stoeva, B. Rachev, Retrieval by Spatial Similarity in Image Databases, Proc. ADBIS'2003, vol.2 121-130, 2003

Финансови изчислителни мрежи

Янка Н. Янакиева

Резюме: В доклада се представя идеята за конструиране на финансови изчислителни системи за управление на риска, базирани на набор от изчислителни блокове (наречени чипове), които се свързват и работят съвместно в топологична изчислителна структура (мрежа). Специфичен синхронизиращ механизъм осигурява правилното разпространение на изчислението в мрежата. Структурата и функционалността на чиповете позволява конструиране на комплексни изчислителни единици (карти), които моделират различни изчислителни процедури за финансови инструменти, необходими при анализа на портфейли и управлението на риска.

Financial evaluation networks for risk management

Yanka N. Yanakieva

Abstract: This paper provides a brief presentation of an idea for constructing financial evaluation systems for use in risk management, based on a set of computation blocks (called Chips), which can be connected to work together in a topological evaluation structure (network). A specific synchronization mechanism ensures correct calculation propagation within the network. The structure and functionality of the Chips is designed to allow construction of complex computational units (cards), which model different evaluation procedures for financial инструментс needed for portfolio analysis and risk management

.

Увод

Финансовият софтуер за анализи и оценки включва множество базови изчислителни процедури (evaluation engines), които обикновено се намират във финасови библиотеки. Тези процедури се извикват от управляващи програми и за подадени параметри връщат желаните резултати (теоретична цена, печалба/загуба, чуствителности, пазарен риск и т.н.) за отделна позиция или портфейл. Управляващите програми работят спазвайки следната последователност: получаване на входни данни от GUI или от базата от данни, извикване на изчислителните процедури в точно определен ред, предаване на параметри и резултати и показване на на резултатите на екрана. Цялостният изчислителен процес може да бъде представен чрез финансова изчислителна мрежа - структура, състояща се от базови и комплексни изчислителни чипове и карти.

1. Изложение

Изчислителният чип преставлява мрежов възел, който има собствена функционалност, определяща неговия тип. Връзките между чипове в рамките на изчислителната мрежа се осъществяват чрез изводи (pins). Изводите могат да бъдат еднопосочни (вход или изход) или двупосочни (вход и изход). Входните изводи представляват аргументи на предавателна функция, която генерира резултат в изходен извод. Множеството от предавателни функции на даден чип дефинира неговата функционалност. Мрежовата структура

се определа от връзките между изводите. Всеки входен извод може да се свърже с един изходен извод. За всеки изходен извод има един или повече входни изводи. Двупосочните изводи позволяват реализация на обратни функции. Основно свойство на чипа е да генерира резултат т.е. да разпространява изчислението. За целта всеки изходен извод има флагова променлива, която показва дали в изходния извод има изчислена стойност.

2.1. Базови финансови чипове

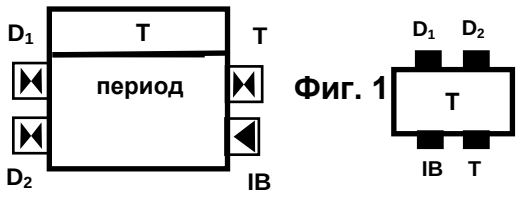
Много финансови изчисления могат да бъдат реализирани като базови финансови чипове, които в зависимост от тяхната функционалност могат да бъдат свързани в мрежа или да участват като компоненти в по-сложни финансови чипове. Ето три примера:

Чип период – определя разликата между две дати в години. Неговата функционалност се определя от следните функции:

$$T = \frac{(D_2 - D_1)_{IB}}{b_{IB}}$$

$$D_2 = (D_1 + T * b_{IB})_{IB}$$

$$D_1 = (D_2 - T * b_{IB})_{IB}$$



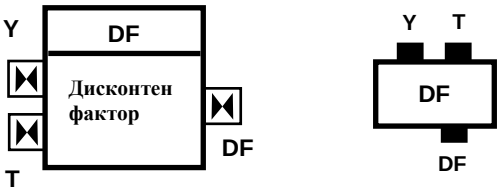
Фиг. 1

Чип дисконтен фактор – неговата функционалност се определя от следните функции:

$$DF = \frac{1}{1 + \frac{Y * T}{100}}$$

$$Y = \left(\frac{1}{DF} - 1\right) * \frac{100}{T}$$

$$T = \left(\frac{1}{DF} - 1\right) * \frac{100}{Y}$$



Фиг. 2

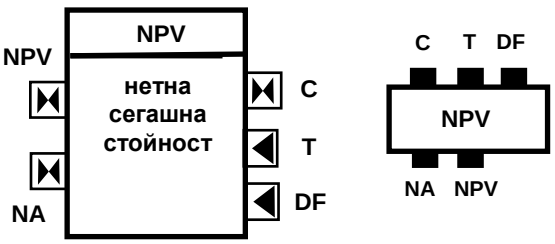
Чип нетна стойност – изчислява нетна настояща стойност на едно плащане, купон, период и дисконтен фактор:

$$NPV = \frac{NA * C * T * DF}{100}$$

$$NA = \frac{NPV * 100}{C * T * DF}$$

$$C = \frac{NPV * 100}{NA * T * DF}$$

$$T = \frac{NPV * 100}{NA * C * DF}$$



Фиг. 3

2.2. Финансова изчислителна мрежа

От финансовите чипове може да се изгради изчислителна мрежа, която моделира определено финансово изчисление. Потребителят може да избира и свързва чипове в съответствие с определено приложение по същия начин, по който конструктурът избира и свързва чипове по предварително зададена схема. Нещо повече, могат да се конструират изчислителни схеми, които нямат аналог сред реалните финансови инструменти.

Свързвайки разгледаните базови финасови чипове и някои аритметични чипове в обща изчислителна структура могат да се моделират изчисления на печалба/загуба, чувствителности, VaR.

Тук се разглежда пример за финансова изчислителна мрежа (фиг. 4), която оценява печалба/загуба (P/L) на паричен заем с два периода на плащане.

Всички необходими входни параметри купон (C), номинал (NA), дати (D_1, D_2, D_0) и лихви (Y_1, Y_2) се подават от външната среда. Изчислението се базира на следните формули:

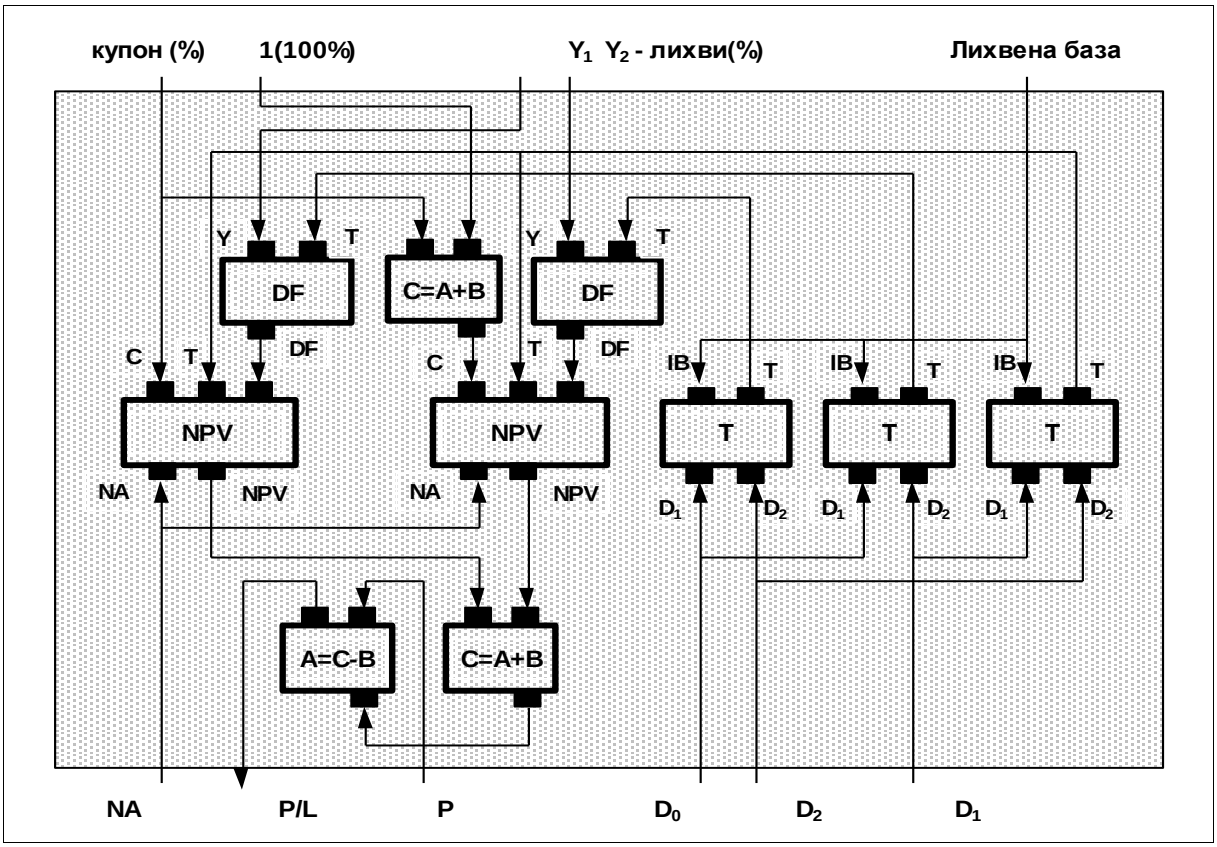
$$P/L = -P + \frac{NA * C * T * DF_1}{100} + \frac{NA * (100 + C) * T * DF_2}{100}$$

където

$$DF_1 = \frac{1}{1 + \frac{Y_1 * T_1}{100}} \qquad DF_2 = \frac{1}{1 + \frac{Y_2 * T_2}{100}};$$

$$T = \frac{(D_2 - D_1)_{IB}}{b_{IB}}$$

$$T_1 = \frac{(D_1 - D_0)_{IB}}{b_{IB}} \qquad T_2 = \frac{(D_2 - D_0)_{IB}}{b_{IB}}$$



Фиг. 4

3. Разпространение на изчислението

Финансовата изчислителна мрежа представлява по принцип множество от комуникиращи крайни автомати, които могат да работят паралелно. Ако всеки чип се стартира като отделен процес (Java thread) работата на мрежата би била твърде неефективна поради необходимостта от често превключване между процесите. За по-голяма ефективност се предлага следният подход: стартиране на подмрежи, пред ставляващи финансови карти, като отделни процеси. Това решение предполага наличие на механизъм за разпространение на изчислението и за управление на подмрежите.

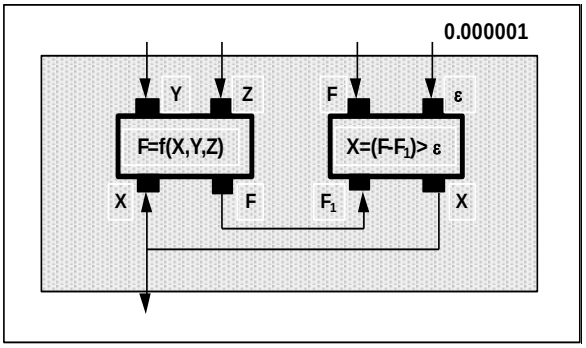
Механизмът за разпростра нние на изчислението може да се релизира чрез управляващ клас, който познава структурата на подмрежата т.е. известни са му връзките между изводите на чиповете в картата. Механизмът работи със стек на събитията, съдържащ неконсумираните събития и референции към съответните чипове. Всеки извод на чип има булева променлива (tag), която сигнализира дали в извода има или няма нова стойност. Процесът на разпространение на изчислението започва с празен стек на събитията и с установяване на стойност **false** във всички булеви променливи. В даден момент външната среда (или друга карта) записва едно или повече събития в стека и подава стойности във входните изводи. В този момент се стартира механизмът за разпространение на изчислението, който консумира събитие от стека, подава стойности и установява булевите променливи на входните изводи на чиповете. За всяко събитие механизмът за разпространение на изчислението прави проверка за наличност на стойности в изводите на даден чип и установява дали може да бъде изчислена някоя от неговите функции. Предавателната функция изпраща резултатите в изходните чипове и установява стойности в техните булеви променливи. В стека на събитията постъпват нови събития. След обработване на едно събитие, механизмът за разпространение на изчислението взима следващото събитие от върха на стека. Разпространението на изчислението завършва, когато стекът се изпразни. По време на разпространението на изчислението се генерират изходни резултати, които се подават на външната среда или на други паралелно работещи карти. Възможно е след завършване на разпространението на изчислението някои от чиповете да не са изчислили своите резултати поради липсващи данни от външната среда. В този случай е необходимо да се извърши синхронизация с други процеси (карти), която ще даде възможност за продължаване на разпространението на изчислението или за стартиране на изчисление в тях. По време на синхронизацията се обменят данни и събития между картите в съответствие с техните интерфейси.

4. Конструиране на по-сложни изчислителни блокове

Примерът от фиг. 4 показва как от множество прости чипове могат да се конструират комплексни изчислителни структури с по-сложна функционалност. Такава структура може да се дефинира като чип от "по-високо ниво", който може да се използва еднократно или многократно в изчислителна карта със сложна изчислителна функционалност – например "чип генератор на дисконтна лихва", чип "рискова стойност на паричен поток, чип "биномиално дърво" и други. Такива чипове могат да включват и итеративни изчисления или изчисления на имплицитни стойности. Последните касаят финансови параметри, които не могат да бъдат определени от изчислителната мрежа

дори и при наличие на необходимите входни данни. За получаване на имплицитни резултати следва да се използва затворена циклична структура, реализирана от чип, който работи като компаратор. Финансови функции, изискващи такива изчисления са вътрешна норма на възвращаемост (IRR) за паричен поток, вътрешна волатилност за опция и т.н.

Фиг. 5 представя модел на чип за изчисление на имплицитни стойности.



Фиг. 5

5. Заключение

Предложеният метод за създаване на финансови приложения от множество базови компоненти заедно с техниките за построяване на по-сложни структури и за управление на изчислението има следните предимства:

- съдържа прост механизъм за разпространение на изчислението в мрежовата структура, който е независим от нейната сложност и е многократно използваем
- дава възможност за гъвкаво и лесно създаване на сложни финансови приложения
- чрез избор и свързване на чипове могат да бъдат конструирани от крайния потребител произволни нестандартизирани финансови инструменти.

Литература

[1] Frank J. Fabozzi, Fixed income Mathematics, Probus Publisching Company, USA, 1993

[2] John Hull, Options, Futures and other derivatives securities, Prentice Hall, INC, 1989

[3] Bruce Eckel, Thinking in Java, 2000, Prentice Hall, INC, 2000

Компютърно-математически метод за синтез на оптимални сингулярни адаптивни компютърни наблюдатели – част II

Любомир Н. Сотиров, Стела Л. Сотирова

Резюме: Предложен е компютърно-математически метод за алгоритмичен синтез на новото поколение адаптивни компютърни наблюдатели, включващи оптимални оценители на вектора на началното състояние, идентификатори на параметрите и оптимални сингулярни (пълни и дегенеративен) адаптивни компютърни наблюдатели на вектора на текущото състояние за линейни стационарни дискретни системи с един вход и един изход. Оценяването на елементите на вектора на началното състояние не е присъщо на методите за синтез на адаптивни наблюдатели, базирани само на идентификатори. Важно е отпадането на процедурата за синтез на системи със зададени полюси, затрудняваща компютърното приложение на резултатите от теорията на наблюдението на Луенбергер и методите за синтез на адаптивни наблюдатели, базирани само на идентификатори. Синтезираният оптимален сингулярен адаптивен компютърен наблюдател има свойства като едрозърнест и естествен паралелизъм, които позволяват неговата софтуерна и хардуерна интерпретация също и с паралелно-векторни компютърни архитектури. Конструираният алгоритъм е интерпретиран и в MATLAB-среда.

A Computer – Mathematical Method for Synthesis of Optimal Singular Adaptive Computer Observers – Part II

Lyubomir N. Sotirov, Stela L. Sotirova

Abstract: A computer- mathematical method for algorithmic synthesis of the new generation of adaptive computer observers, including optimal estimators of the initial state vector, identifiers of the parameters and optimal singular (full and degenerate) adaptive computer observers of the current state vector for single-input single-output linear stationary discrete systems is suggested. The estimation of the initial state vector elements is not typical of the methods for synthesis of adaptive observers, based only on identifiers. The important fact is the dropping off of the procedure for the synthesis of systems with given poles, hindering the computer application of the results from Luenberger's observation theory and the methods for synthesis of adaptive observers, based only on identifiers. The synthesized optimal singular adaptive computer observer has features such as large- scale granularity and natural parallelism, which also allow its software and hardware implementation on a parallel- vector computer architecture. The constructed algorithm is implemented in MATLAB- environment.

3. За оптималността, сингулярността и робастността на адаптивното наблюдение с оценяване на началния вектор на състоянието

Ако означим грешката на наблюдението с

$$\mathbf{e}(k)=\mathbf{x}(k)-\hat{\mathbf{x}}(k), \quad k=0, 1, 2, \ldots, \tag{8}$$

може да се покаже, че тя удовлетворява векторно-матричното уравнение от вида:

$$\mathbf{e}(k+1)=\hat{\mathbf{F}}\,\mathbf{e}(k), \tag{9}$$

$$\mathbf{e}(0)=\mathbf{x}(0)-\hat{\mathbf{x}}(0),$$

$$k=0, 1, 2, \ldots,$$

ако оценките на векторните параметри са точни, т.е. ако:

(10) $\hat{\mathbf{a}} = \mathbf{a}, \qquad \hat{\mathbf{b}} = \mathbf{b},$

Общото решение на *(9)* може да бъде записано като:

(11) $\mathbf{e}(k) = \hat{\mathbf{F}}^{\mathbf{k}}\mathbf{e}(0), \qquad k=1, 2, \ldots,$

От тук следва, че нормата на грешката удовлетворява съотношението:

(12) $\left\|\mathbf{e}(k)\right\| = \left\|\hat{\mathbf{F}}^{\mathbf{k}}\mathbf{e}(0)\right\|, \qquad k=1, 2, \ldots,$

от където може да запишем:

(13) $\left\|\mathbf{e}(k)\right\| \le \left\|\hat{\mathbf{F}}^{\mathbf{k}}\right\|\left\|\mathbf{e}(0)\right\|, \qquad k=1, 2, \ldots$

Очевидно е, че се разполага с две независими средства, за да се оптимизира по бързодействие процеса на адаптивно наблюдение:

1) Матрицата $\hat{\mathbf{F}}$ трябва да бъде синтезирана така, че за минимален брой стъпки да се изпълнява условието:

(14) $\left\|\hat{\mathbf{F}}^{\mathbf{k}}\right\| \to 0, \qquad k=1, 2, \ldots$ Известно е, че за разлика от случая на непрекъснати системи, грешката на наблюдението на текущия вектор на състоянието по Luenberger може да се направи да затихне за ***n*** на брой стъпки, като вектора **g** се синтезира така, че собствените стойности на $\hat{\mathbf{F}}$ да бъдат равни на нула. Тогава:

(15) $\mathbf{e}(n) = \hat{\mathbf{F}}^{\mathbf{n}}\mathbf{e}(0) = 0,$

за произволен начален вектор **e***(0)*.

Условието *(15)* означава, че матрицата $\hat{\mathbf{F}}$ е нилпотентна (т.е. има нулеви собствени стойности) и в този случай дискретния модел на грешката *(9)* е с безкрайна степен на устойчивост.

2) Очевидно е, че ако синтезираната оценка на началния вектор на състоянието е точна, т.е. ако:

(16) $\hat{\mathbf{x}}(0) = \mathbf{x}(0),$

то получава своя минимум квадратичният функционал от вида:

(17) $\left\|\mathbf{e}(k)\right\|^2 = \left\|\mathbf{x}(k) - \hat{\mathbf{x}}(k)\right\|^2,$

$k=0, 1, 2, \ldots,$

където нормата е дефинирана като:

(18) $\left\|\mathbf{e}(k)\right\| = \sqrt{\left\langle \mathbf{e}(k), \mathbf{e}(k) \right\rangle},$

$k=0, 1, 2, \ldots,$

където със символа $\langle(\cdot),(\cdot)\rangle$ е означено скаларното произведение на два вектора.

От гореизложеното следва, че ако се изпълнява *(16)*, то грешката на наблюдението моментално ще се нулира, независимо от характера на разпределението на собствените стойности на матрицата $\hat{\mathbf{F}}$, т.е.:

(19) $\mathbf{e}(k)=0,$ за всички

$k=0, 1, 2, \ldots$

Очевидно, адаптивният подход (на базата на оценяването на началния вектор на състоянието), поражда нови възможности, свързани с разкритата робастност на матрицата $\hat{\mathbf{F}}$ на ОСА

наблюдателя към разположението на нейните собствени стойности. Това позволява да се оптимизира по бързодействие наблюдението на текущото състояние по Luenberger, което от своя страна води до частния (сингулярен) случай на точните наблюдения и позволява да се избегне междинното решение на задачата за синтез при зададени полюси, което е съпроводено с редица усложнения при компютърна интерпретация.

От друга страна, анализът на иновационната добавка в математическия модел на пълния ОСА наблюдател показва, че ако:

(20) $\mathbf{y}(k)=\mathbf{c}^{\mathrm{T}}\hat{\mathbf{x}}(k)$,

$k=0, 1, 2, \ldots$,

то естествено се преминава от пълен ОСА наблюдател към дегенеративен.

Очевидно, необходимостта от използването на пълния или дегенеративния ОСА наблюдател ще се определя и от изискванията на потребителя за съответно гарантирана точност на решението на задачата за ОСА наблюдение и от друга страна, от простата за реализация структура на дегенеративните ОСА наблюдатели.

4. MATLAB- програмиране на M1A6- алгоритъм за ОСА наблюдение

Стъпка 1. Формират се вектори и матрици от входно- изходни данни:

```
i=1:n;           y1=[y(i)]';
i=n+1:2*n;       y2=[y(i)]';
i=2*n+1:3*n;  y3=[y(i)]';
for j=1:n-1
  for q=1:n-j
    i=j+q;
    U(i,j)=u(q);
  end
end
Utr=[U,zeros(n,1)];
for i=1:n
  for m=i:i+n-1
    j=n-m+i;
    U11(i,j)=u(m);
  end
end
for i=1:n
  for m=i+n:i+2*n-1
    j=2*n-m+i;
    U21(i,j)=u(m);
  end
end
for i=1:n
  for j=1:n
    Y12(i,j)=y(i+j-1);
  end
end
for i=1:n
  for j=1:n
    Y22(i,j)=y(i+j-1+n);
  end
end
OMEGA=[U11, Y12; U21, Y22];
Y=[y2;y3];
```

където **Utr**, **U11** и **U21** са *n*x*n* матрици на Теплиц;
Y12 и **Y22** са *n*x*n* матрици на Ханкел.

Стъпка 2. Оценката TITAe на вектора

$$\hat{\Theta}^T = \left[\hat{h}^T : \hat{a}^T\right],$$

се изчислява, както следва:

```
TITAe= OMEGA\Y;
i=1:n;      he= [ TITAe(i)];
i=n+1:2*n;  ae= [TITAe(i)];
```

Стъпка 3. Оценката be на вектора **b** се изчислява със следния модул:

```
T=eye(n);
for j=1:n-1
  for m=j+1:n
    i=n-m+j+1;
    T(i,j)=-a(m);
```

```
  end
end
be=T\he;
```

Стъпка 4. Началният вектор на състоянието **xe0** се изчислява, използвайки оптималния оценител от вида:

```
xe0= y1-Utr*be;
```

Стъпка 5. Текущият вектор на състоянието **xe(:, k)** се оценява с помощта на следния пълен оптимален сингулярен адаптивен наблюдател:

```
I=eye(n-1);
NUL=zeros(n-1,1);
Ae=[NUL, I; ae'];
Fe=Ae-g*c';
xe(:,1)=xe0;
for k=1:3*n-1
  xe(:,k+1)=Fe*xe(:,k)+be*u(k,:)+g*xe(1,k);
end
```

където избора на елементите на **g**-вектора е произволен и е препоръчан погоре.

5. Компютърни експерименти

На базата на симулирани данни, например, за дискретна система от 4-ти ред, чиито векторни параметри и начално състояние са:

$$\mathbf{A}=\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -0.75 & 0.97 & -0.3 & 1.2 \end{bmatrix},$$

$$\mathbf{b}=\begin{bmatrix} 1.5 \\ 1.7 \\ 0.9 \\ 1 \end{bmatrix},$$

$$\mathbf{x}(0)=\begin{bmatrix} 1.5 \\ 1.7 \\ 1.9 \\ 1.1 \end{bmatrix},$$

като се използва следния MATLAB- модул:

```
u=[0.1, 0.2, 0.3, 0.5, 0.4, 0.3, 0, -0.2, -0.3,
-0.4, -0.5, -0.3]';
n=4;
a=[-0.75; 0.97; -0.3; 1.2];
I=eye(n-1);
NUL=zeros(n-1,1);
A=[NUL, I; a'];
b=[1.5; 1.7; 0.9; 1];
x0=[1.5; 1.7; 1.9; 1.1];
c=[1; zeros(n-1,1)];
x=ltitr(A,b,u,x0)';
k=1:3*n;
y(k)=c'*x(:,k);
```

може да се симулира изходния сигнал:

```
y=[1.5000  1.8500  2.3700  1.9800
2.8140  3.8322  3.6550  3.6625
3.9882  3.6208  3.5278  3.9709].
```

В резултат са получени съответните оценки, със следните относителни грешки:

$$\frac{\|\mathbf{a}-\mathbf{ae}\|}{\|\mathbf{a}\|}=7.5974.10^{-15},$$

$$\frac{\|\mathbf{b}-\mathbf{be}\|}{\|\mathbf{b}\|}=1.3720.10^{-14},$$

използвайки идентификатора.

При това

cond(Ω)=*523.2682*,

cond(T)=*9.2855*

и

$$\mathbf{he}=[1.5,\ -0.1,\ -0.69,\ -1.025]^{\mathrm{T}}.$$

С помощта на оптималния оценител на началното състояние са получени оценки със следната относителна грешка:

$$\frac{\|\mathbf{x0}-\mathbf{xe0}\|}{\|\mathbf{x0}\|}=0.0165.10^{-13}.$$

Дегенеративният ОСА наблюдател оценява текущия вектор на състоянието със следните относителни грешки:

$$\frac{\|\mathbf{e}(1)\|}{\|\mathbf{x}(1)\|} = 0.0393.10^{-13},$$

$$\frac{\|\mathbf{e}(2)\|}{\|\mathbf{x}(2)\|} = 0.0726.10^{-13},$$

$$\frac{\|\mathbf{e}(3)\|}{\|\mathbf{x}(3)\|} = 0.1071.10^{-13},$$

…,

$$\frac{\|\mathbf{e}(10)\|}{\|\mathbf{x}(10)\|} = 0.2477.10^{-13},$$

$$\frac{\|\mathbf{e}(11)\|}{\|\mathbf{x}(11)\|} = 0.2335.10^{-13}$$

където

$\mathbf{e}(k)=\mathbf{x}(k)-\mathbf{xe}(k)$, $k=1, 2, ...$

Неговите полюси се разпределят както следва:

$$\mathrm{eig}(\mathbf{Fe}) = [0.7516,\quad 1.1873,\quad -0.3695+0.8390i,\quad -0.3695-0.8390i].$$

Ако пълния ОСА наблюдател (при **g**=**be**) , със следните полюси:

$$\mathrm{eig}(\mathbf{Fe}) = [1.5094,\quad -1.3052,\quad -0.2521+0.6399i,\quad -0.2521-0.6399i,]$$

се използва, порядъка на оценяване се съхранява същия:

$$\frac{\|\mathbf{e}(1)\|}{\|\mathbf{x}(1)\|} = 0.0393.10^{-13},$$

$$\frac{\|\mathbf{e}(2)\|}{\|\mathbf{x}(2)\|} = 0.0726.10^{-13},$$

$$\frac{\|\mathbf{e}(3)\|}{\|\mathbf{x}(3)\|} = 0.1064.10^{-13},$$

…,

$$\frac{\|\mathbf{e}(10)\|}{\|\mathbf{x}(10)\|} = 0.2463.10^{-13},$$

$$\frac{\|\mathbf{e}(11)\|}{\|\mathbf{x}(11)\|} = 0.2315.10^{-13}.$$

$\mathbf{w}^{\mathbf{T}} = [0,\quad 0,\quad 0,\quad 0]$,

$\mathbf{g}^{\mathbf{T}} = [1.2,\quad 1.14,\quad 1.978,\quad 2.4456]$

са подобни:

$$\frac{\|\mathbf{e}(1)\|}{\|\mathbf{x}(1)\|} = 0.0087.10^{-12},$$

$$\frac{\|\mathbf{e}(2)\|}{\|\mathbf{x}(2)\|} = 0.0179.10^{-12},$$

$$\frac{\|\mathbf{e}(3)\|}{\|\mathbf{x}(3)\|} = 0.0291.10^{-12},$$

…,

$$\frac{\|\mathbf{e}(10)\|}{\|\mathbf{x}(10)\|} = 0.1243.10^{-12},$$

$$\frac{\|\mathbf{e}(11)\|}{\|\mathbf{x}(11)\|} = 0.1374.10^{-12}$$

Резултатите от оценяването на пълния ОСА наблюдател при желани нулеви полюси, т.е. при

а техните истински полюси са следните:

$$\mathrm{eig}(\mathbf{Fe}) = [0.1425,\quad -0.1426,\quad 0.0001+0.1425i,\quad 0.0001-0.1425i].10^{-3}.$$

Характеристиките на третия пълен ОСА наблюдател, при желани полюси от вида:

$\mathbf{w}^{\mathbf{T}} = [0.1,\quad 0.1,\quad 0.1,\quad 0.1]$,

$\mathbf{g}^{\mathbf{T}} = [0.8,\quad 0.72,\quad 1.59,\quad 1.7181]$

са подобни на предходните наблюдатели, т.е.

$$\frac{\|\mathbf{e}(1)\|}{\|\mathbf{x}(1)\|} = 0.0900.10^{-12},$$

$$\frac{\|\mathbf{e}(2)\|}{\|\mathbf{x}(2)\|} = 0.0184.10^{-12},$$

$$\frac{\|\mathbf{e}(3)\|}{\|\mathbf{x}(3)\|} = 0.0299.10^{-12},$$

…,

$$\frac{\|\mathbf{e}(10)\|}{\|\mathbf{x}(10)\|} = 0.1281 \cdot 10^{-12},$$

$$\frac{\|\mathbf{e}(11)\|}{\|\mathbf{x}(11)\|} = 0.1419 \cdot 10^{-12}$$

при това неговите полюси не съвпадат с желаните полюси, т.е.

$$\text{eig}(\mathbf{Fe}) = [0.0998,\ \ 0.1000 + 0.0002i,\ \ 0.1000 - 0.0002i,\ \ 0.1002].$$

Ако се използва четвъртия пълен ОСА наблюдател (при **g=he**), със следното разпределение на полюсите:

$$\text{eig}(\mathbf{Fe}) = [1.5094,\ \ -1.3052,\ \ -0.2521 + 0.6399i,\ \ -0.2521 - 0.6399i]$$

порядъкът на оценяване остава същия,т.е.

$$\frac{\|\mathbf{e}(1)\|}{\|\mathbf{x}(1)\|} = 0.0382 \cdot 10^{-13},$$

$$\frac{\|\mathbf{e}(2)\|}{\|\mathbf{x}(2)\|} = 0.0702 \cdot 10^{-13},$$

$$\frac{\|\mathbf{e}(3)\|}{\|\mathbf{x}(3)\|} = 0.1053 \cdot 10^{-13},$$

…,

$$\frac{\|\mathbf{e}(10)\|}{\|\mathbf{x}(10)\|} = 0.2521 \cdot 10^{-13},$$

$$\frac{\|\mathbf{e}(11)\|}{\|\mathbf{x}(11)\|} = 0.2389 \cdot 10^{-13}.$$

Относителните грешки са изчислени със следния MATLAB- модул:

```
Na=norm(a-ae)/norm(a);
Nb=norm(b-be)/norm(b);
for k=1:3*n
  Nx(:,k)=norm(x(:,k)-xe(:,k))/norm(x(:,k));
End
```

Резултатите от многобройни компютърни експерименти със симулирани и реални данни от технологически процеси, морски и други подвижни обекти, биотехнологични и електромеханични системи, от икономиката и финансите са с висока гарантирана точност и илюстрират математическата и софтуерна състоятелност на предложения компютърно- математически метод и синтезиран алгоритъм.

6. Заключение.

Входните данни за алгоритъма се формират като матрици на Теплиц и Ханкел, които имат някои специални свойства. В резултат на това, ние може да синтезираме бързи алгоритми.Специфичните свойства могат да бъдат използвани при обръщането на указаните матрици и при директното решаване на линейни системи от алгебрични уравнения със специална плътна структура.Това разширява областите на приложение на предложения компютърно- математически метод.

Като използваме конструирания алгоритъм ние може да оценяваме елементите на началния вектор на състоянието, което не е присъщо на алгоритмите, базирани на теорията на наблюдението на Луенбергер и на методите за синтез на адаптивни наблюдатели, базирани само на идентификатори.

Подобното качество на наблюдението, осъществявано от 5- те ОСА компютърни наблюдатели, разгледани по- горе, очевидно, прави възможно отпадането на процедурата за синтез на системи при зададени полюси, която затруднява компютърното приложение на резултатите от теорията на наблюдението на Луенбергер и на методите за синтез на адаптивни наблюдатели, базирани само на идентификатори.

Алгоритъмът представлява линейна компютърна процедура, на която не са присъщи трудностите, съпътстващи нелинейните алгоритми за съвместна оценка на параметри и състояния.

Нашият алгоритъм има свойства като едрозърнест и естествен паралелизъм, което позволява неговата хардуерна и софтуерна интерпретация още и с паралелно- векторни компютърни архитектури.

Теорията за ОСА наблюдение и моделиране даде възможност за развитие на новото поколение адаптивни компютърни контролери / оптимални, модални, апериодични /бързи/, приспособяващи се към смущения/.

ЛИТЕРАТУРА

13. Sotirov, L. N., 1994, A direct method for adaptive observation of single-input single-output linear stationary discrete systems with initial state vector estimation, Reports of the Bulgarian Academy of Sciences, vol. 47, No. 11, 5-8, Sofia.

14. Sotirov, L. N., 1997, Optimal singular adaptive observation of stationary discrete systems with initial state vector estimation. *Automation and Remote Control, Russian Academy of Sciences, No 9, 110-118, Moscow, New York, London .*

15. Sotirov, L. N., 1997, An algorithm for synthesis of optimal singular adaptive observers with direct estimation of the initial state vector. *International Journal of Systems Science, vol. 28, No 6, 559-562.*

16. Sotirov, L. N., E. G. Sukhov, 1997, A parallel direct method for optimal adaptive estimation of discrete linear system, *Automation and Remote Control, Russian Academy of Sciences, No 10, 154-163, Moscow, New York, London .*

17. Sotirov L.N., 1999, Reduced order optimal singular adaptive observation for a class of discrete systems, *Automation and Remote Control, Russian Academy of Sciences, No.2, 75-82, Moscow, New York, London*.

18. Sotirov L.N., 1999, An algorithm for synthesis of discrete reduced order optimal singular adaptive observers. *International Journal of Systems Science* , vol.30, No. 6, 665-671.

19. Sotirov L.N., 2000, Optimal singular adaptive observation of multi- input discrete linear systems with perturbation, *Automation and Remote Control, Russian Academy of Sciences, No 7, 120-130, Moscow, New York, London .*

20. Sotirov L.N., 2001, An algorithmic synthesis of discrete optimal singular adaptive observers for multiple- input single- output linear systems, . *International Journal of Systems Science* , vol.12, No. 5, 545-551.

21. Sotirov L.N., 2001, An algorithm for synthesis of discrete time- optimal adaptive controllers, Reports of the Bulgarian Academy of Sciences, vol. 54, No. 3, 19-24, Sofia.

22. Sotirov L.N., S.L. Sotirova, 2002, An algorithmic synthesis of discrete optimal singular adaptive observers for single- input single- output linear systems, Reports of the Bulgarian Academy of Sciences, vol. 55, No. 7, 31-36, Sofia.

23. Сотиров Л. Н., Избрани глави от съвременната теория на управлението, 1998, Технически университет, Варна.

24. Сотиров Л. Н., Теория на автоматичното управление, Част 2 / Теория на дискретните системи за автоматично управление /, 2000, Технически университет, Варна.

АПРИОРЕН АЛГОРИТЪМ ЗА РЕИНЖЕНЕРИНГОВА КЛЪСТЕРИЗАЦИЯ

М.Митев В.Божикова

Резюме: В статията е въведена обобщена формализация на проблема за реинежерингова клъстеризация на програмни системи, базираща се на теория на графите. Предложен е теоретично обоснован априорен алгоритъм за декомпозиция на крайни ориентирани графи. Алгоритъмът е аналитично изследван. Посредством моделни изпитания е доказана неговата ефективност

APRIOR ALORITHM REGARDING THE RE-ENGINEERING CLUSTERIZATION

M.Mitev V.Bojikova

Abstract:**:** The publication introduces a common formalization of the problem regarding the re-engineering clusterization of programs systems based on the theory of graphs. A theory based aprior algorithm for decomposition of edged oriented graphs is suggested. The algorithm is investigated by the analytical approach. The effectiveness of the algorithm is proved by the means of the modeling testing.

Увод

Интензивното развитие на софтуерното производство и общосистемното програмно осигуряване определят процес на непрекъснато разработване на нови модификации и версии на използваните програми. Този процес е нискоефективен поради отсъствието на пълна проектна доку-ментация. Разработените спецификации не отговарят на възможностите на новите програмни платформи. В определени случай разходите по създаването на нови версии е икономически неиздържано, независимо от функционалната пригодност на програмата

Реинженерингът е сравнително нова дейност свързана с адаптирането на програмно осигуряване със значителна стойност към променените условия на апаратната и програмна част. Счита се [1,2], че постигането на тази цел е изгодно, ако разходите свързани с реинженеринга не превишават 50% от разходите по създаването на нова версия

Независимо от постигнатите резултати [3-6] не съществува единна концепция за технологично осигуряване /методическо и инструментално/ за реинженерингова дейност. Съществуват множество разработки, които се основават на отделни, частни постановки и решават специфични проблеми.

Настоящият доклад е ориентиран към клъстеризацията на програмното осигуряване, като едно от средствата за неговото преструктуриране и адаптиране към новите условия на програмната и информационна среда.

1. Формализация

В условията на непълна документация и разработени спецификации, които са съобразени с вече несъществуващи или често непод-

държани програмни условия, формализацията може да бъде сведена до следното:

- функционално разделяне на програмния код на отделни атомарни елементи. Всеки елемент представлява относително самостоятелна, неделима програмна част, осигуряваща изпълнението на определена функция. Обемът на атомарните елементи, тяхното кодово съдържание и изпълнявани функции зависи от нивото на абстракция */управляващи, изчислителни и обработващи/*. *Управляващите* атомарни елементи са пряко свързани с общосистемното програмно осигуряване и взаимодействуват с негови функции. *Изчислителните* изпълняват непосредствено преобразуване на информацията и практически не зависят от използваната програмна платформа. *Обработващите* атомарни елементи включват входно – изходни операции и преструктуриране на информацията. Те са свързани с програмното и косвено с апаратното осигуряване на изчислителната конфигурация.
- В процеса на изпълнение, отделните атомарни елементи се свързват и взаимодействуват по между си.. Това се регламентира от изпълнението на определени функции, от стойността на междинни резултати, от прякото участие на потребителя и от управляващото въздействие на операционната система
- При тази постановка формализацията на проблема за реинженерингова клъстеризация се свежда до представянето на атомарните елементи като върхове $X=\{x_i\}$, $N = |X|$ на краен, ориентиран граф $G(X,U)$. Функционалната или програмна връзка между елементите може да се интерпретира като множество от дъги $U = \{u_{ij}\}$, свързващи върховете на графа

За отчитане на *обемните и честотни характеристики* на атомарните елементи са въведени следните множества

$W=\{w_i\}$ – тегловна интерпретация на върховете,

$V=\{v_{ij}\}$ – честотна интерпретация да дъгите.

Следователно съществува едно – еднозначно съответствие между елементите на X и W, и U и V.

При тази постановка клъстеризацията се свежда до целево декомпозиране на графа G на множество от подграфи $G = \{g_m\}$, където $m=1,2,….M$ е броят на клъстерите. Всеки от определените клъстери е тегловно ограничен от константата W_0, отчитаща наложени ограничения в динамиката на клъстерите

Нека с $R=\{r_k\}$ се обозначи множеството на всички възможни решения за декомпозиция на графа при ограничително условие W_0. С r_0 се обозначава начално, в общия случай произволно решение, с r_e – екстремално /оптимално/ и с r_p – рационално решение. Всяко от посочените решения се определя от съдържанието на отделните подграфи /клъстери/ и от вътрешните , респ. външни дъги.

По определение, екстремално решение r_e е това при което сумата от теглата на външните дъги е минимална, респ. за вътрешните – максимална т.е.

$$(1) \quad f_e = \sum_{m=1}^{M} \sum_{\forall x_i, x_j \in Xm}^{M} v_{ij} = min$$

За рационално решение r_p се счита това, при което е достигнат локален екстремум или е изразходвано предварително лимитирано време за клъстеризация.

Следователно функция (1) може да се приеме като *целева* при ограничението W_0.

2.Априорен алгоритъм

В съвременните програмни системи връзката между отдерните атомарни елементи е силно изразена. Това се дължи на повишената динамика на активиране от страна на потребителя, на операционната система и на новите възможности в резултат от обектното и събитийно програмиране.

Следователно, допускането, че графа *G* интерпретиращ съвременна програмна система ще се доближава до *пълен граф* е основателна.

На тази постановка е базиран разработения априорен алгоритъм. При него честотните характеристики на дъгите не са известни априорно, но се приемат , че те са с висок интезитет и са приблизително еднакви , т.е.

$(\forall u_{ij} \in U) \rightarrow v_{ij} = const,$

в частност $v_{ij} = 1$

Нека графа *G* е зададен с матрицата на свързаност с размерност *N x N.*

Решението $r_e \in R$ в класа *M* определя стойност на целевата функция

$$(2) \qquad f_e = \sum_{m=1}^{M} n_m^2$$

Без да се нарушава общността на разглежданията може да се приеме, че подграфите са индексирани в низходящ ред на техните мощности., т.е.

$$(3) \qquad n_1 \geq n_2 \geq n_3 \geq \ldots\ldots\ldots \geq 1$$

В сила е следното твърдение:

Решението $r_e \in R$ *определя* $f_e \geq f_p$ *в класа M, ако е получено посредством последователен процес , при които на всяка стъпка се получава максимален по мощност подграф при спазване на ограничителното условия*

$$(4) \qquad \sum w_i \leq W0 \quad \forall x_i \in X_m$$

Допуска се противното $f_e < f_p$. В частност нека подграфа $g_2 \in G$ може да увеличи своята мощност за сметка на $g_1 \in G$ т.е.

$$(5) \qquad n_1 = n_1 - 1 \; ; \; n_2 = n_2 + 1$$

$$(6) \; f_p = (n_1 - 1)^2 + (n_2 + 1)^2 + \sum_{m=3}^{M} n_m^2$$

От допускането следва, че

$$n_1^2 + n_2^2 + \sum_{m=3}^{M} n_m^2 < (n_1 - 1)^2 + (n_2 + 1)^2 + \sum_{m=3}^{M} n_m^2$$

$$2n_2 - 2n_1 + 2 > 0$$

$$n_2 > n_1 - 1$$

което противоречи на (**3**)

За да се постигне зададената стойност на M /ако това е възможно/ или да се сведе до възможния минимум, то е необходимо отделяните подграфи да бъдат тегловно За да се постигне зададената стойност на *M* /ако това е възможно/ или да се сведе до уплътнявани

За целта, нека G_d е допълнението на отделеното подмножество от графи $\{ g_m \}$ до графа *G.* За $g_m \in G$ съществува тегловната разлика

(7) $\Delta w = W_0 - \sum_{\forall x_i \in X_m} wi$

Ако върховете на $X_m \subset X$ и $X_d \subset X$ са сортирани във възходящ ред на техните тегла

(8) $(\forall x_i\, x_j \in X_m)\,(i < j) \rightarrow (w_i \leq w_j)$
$(\forall x_i\, x_j \in X_d)\,(i < j) \rightarrow (w_i \leq w_j)$

то следователно

(9) $(\forall x_i \in X_m)\,(\exists x_j \in X_d) \rightarrow (w_i + \Delta w \leq w_j)$

съществуват двойки върхове , с които може да се извърши тегловно уплътняване на отделяния подграф. Това позволява в подредените подмножества от върхове да се определят долна граница в X_m и съответно горна граница в X_d , в диапазона на които е възможно тегловно уплътняване. Определянето на граници намалява броя на проверките.

На основание доказаното твърдение и определените граници за тегловно уплътняване на подграфите се предлага следния априорен алгоритъм за реинженерингова клъстеризация на програмни системи

.

1. Формиране на краен ориентиран граф G(X,U) на базата на определените атомарни елементи .
2. Проверка за коректност на получения граф /изолирани върхове, подграфи и др./
3. $m=1$
4. $X_d=X$, сортиране на върховете на X_d във възходящ ред на теглата им.
5. Отделяне на максимален по мощност подграф на базата на първите n_m върхове, които не нарушават ограничителното условие (4).
6. Определяне на долната и горна граници и уплътняване на подграфа.

———

7. $m = m + 1;\ X = X \setminus \{ g_m \}$
8. Ако $IXI > 0$, то се изпълнява т. 4

Разработеният априорен алгоритъм се основава на силната свързаност на графа. Същият е ниско ефективен при програми с ясно изразен процедурен характер.

Многократното прилагане на алгоритъма, отначало върху атомарни елементи , а в последствие върху определените клъстери при нарастващи стойности на ограничителното условие води до получаването на нова структура на разглежданата програмна система.

3. Аналитични изследвания на алгоритъма.

Бързодействието на алгоритъма се определя от максималното количество на операциите от типа *преместване*, *събиране* и *сравнение*. Оценката трябва да е функция на мощността на разглеждания граф. За целта се определя максималната горна и долна граници на операциите във функция на броя на атомарните елементи /клъстери/.

Разпределението на операциите по точки от алгоритъма е следното:
m.2 - изпълнява се еднократно. Количеството на операциите зависи от използваните методи за откриване на изолирани върхове и подграфи.
m.4. – изпълнява се $M - 1$ пъти при непрекъснато намаляващ брой на върховете. Количеството на операциите зависи от използвания метод за сортиране. Теоретичната долна граница на операциите от типа *сравнение* съгласно [Кнут , том 3] е $N(\log_2 N – 1.43)$. Горната граница се изменя в широк

диапазон, от $N(N-1)/2$ при *bubble sort* до $N \log_2 N$ при метода на сливането.

m.5 – *N* на брой операции от типа *събиране* и същия брой операции *сравняване*.

m.6. – за определяне на долната и горна граници са необходими не повече от N^2 операции от типа *сравняване* и *2N* от типа събиране.

Останалите точки са с несъществен брой операции.

Ако в т.4 се използва метода *bubble sort* за сортиране при най-тежкия случай – подграфи с приблизително еднакъв брой върхове, то общото количество операции от типа сравнение ще се определи от функцията

С отчитане на (3) и (4), съотношение (2) се записва окончателно, като

$$(10) \quad RF(N) = N/M * N(N-1)/2 + N^2$$

$$RF(N) = A*N^3 + B* N^2 ;$$

$$A= 1/2M \text{ и } B = 1- 1/2M$$

За $RF(N)$ съществуват функциите $O(N^3)$ и $o(N^4)$ за които вида:

$$\lim_{N \to \infty} \frac{RF(N)}{O(N^3)} = \frac{AN^3 + BN^2}{O(N^3)} = A = const$$

$$\lim_{N \to \infty} \frac{RF(N)}{o(N^4)} = \frac{AN^3 + BN^2}{o(N^4)} = 0$$

Следователно, разработеният априорен алгоритъм за реинженерингова клъстеризация с голяма степен на достоверност може да се счита, че е от класа на полиномиалните

4.Експерименти

За изследване на разработения алгоритъм е създадена експериментална среда съставена от генератори на коректни структури, генератори симулиращи произволното активиране на атомарни елементи с установена свързаност по между си, програмна част за измерване на необходимото процесорно време за клъстеризация и програми за статистическа обработка на получените резултати.

Проведени са моделни експерименти за реинженерингова клъстеризация на повече от 30 генерирани стуктури при запълване над 70 % на матрицата на свързване.

5. Изводи

Резултатите от проведените експерименти на моделно генерирани структури потвърждават работоспособността на предложения алгоритъм. Получените резултати са удовлетворителни по отношение на минималната външна свързаност на клъстерите. Изразходваното време за клъстеризация е от порядъка на аналитичните резултати.

Априорният алгоритъмът може да намери приложение в качеството на инструментално средство за клъстеризация при условията на силна свързаност на отделните клъстери. На основание получените резултати от клъстеризацията след априорния алгоритъм могат да се прилагат и апостериони алгоритми за клъстеризация, при получена допълнителна информация за честотната активност на клъстерите и при отчитане специфичните особености на операционната среда

След подходяща адаптация алгоритъмът може да се използва при разработване на проектни решения за изграждане и поддържане на информационна среда, при проектиране на

активен потребителския интерфейс и не на последно място като компонент на технология за реинженерингова клъстеризация на програмни системи..

Литература:

[1] Н.Васильева Реинжиниринг программного обеспечения: определения, стратегии, экономическое обоснование, Спецсеминар "Реинжиниринг программного обеспечения", 01.03.2001: http://se.math.spbu.ru/courses/reeng/

[2] STSC (Software Technology Support Center), Reengineering Technical Report, Volume 1, Hill AFB, Utah /US, Oct. 1995

[3] Терехов А.А., Верхуф К.,Проблемы языковых преобразований, Открытые системы, №5-6, 2001. С. 54–62

[4] Terekhov A.A., Verhoef C., The Realities of Language Conversions, IEEE Software, Vol. 17, No. 6, 2000. P. 111–124

[5] Терехов А.Н., Эрлих Л., Терехов А.А., Перспективы реинжиниринга, Компьютер-Пресс, №11, 1999. С. 124–127

[6] Ira D. Baxter, “Using Automated Transformation Systems for Software Maintenance and Reengineering” - ICSM’2001 – Proceedings of the IEEE International Conference on Software Maintenance – 6-10.11.2001, Florence, Italy

-