Trendy v podnikání, 4() 11-18

ISSN 1805-0603
Publisher: UWB in Pilsen

OPTIMALIZÁCIA PROCESOV POMOCOU EVOLUČNÝCH ALGORITMOV

Jana Filanová

ÚVOD

Súčasná doba si žiada stále rozsiahlejšie a zložitejšie procesy, ktoré sú neoddeliteľnou súčasťou podnikov, rôznych organizácií a v neposlednom rade aj nášho každodenného života. Preto je veľkou výhodou, ba dokonca nutnosťou tieto procesy optimalizovať, aby sa dosiahla ich väčšia prehľadnosť a hlavne lepšia funkčnosť procesu samotného. Optimalizácia procesov prináša zlepšenie funkčnosti jednotlivých činností skladajúcich sa z rôznych optimalizovaných podprocesov, prípadne na tej najvyššej úrovni zlepšenie celkovej štruktúry a zvýšenie výkonnosti jednotlivých firiem a organizácií. Ďalším dôvodom, prečo optimalizovať, resp. zlepšovať a skvalitňovať jednotlivé procesy, je potreba tento proces sprehľadniť a prispôsobiť novým podmienkam trhu alebo spoločnosti, ktoré sa neustále vyvíjajú. Optimalizácie procesov sú pre firmy či iné organizácie dôležité aj z toho dôvodu, že im poskytujú detailný prehľad o procesoch samotných, môžu im odhaliť chyby v technológii výroby alebo pracovných či iných postupoch a často môžu podnikom ušetriť vysoké finančné náklady.

1. MODELOVANIE PROCESOV

Súčasťou analýzy procesov je procesné mapovanie, t. j. vytváranie procesných máp (modelov). Procesná mapa zobrazuje vstupno-výstupné vzťahy procesov, aktivít a útvarov. Pomocou postupnosti procesných krokov sú zdokumentované aktivity nutné k transformácii vstupov na výstupy. Pomocou procesného mapovania je možné identifikovať kritické rozhrania, časové prekrývania podprocesov, prípadne slabé miesta (nelogické, chýbajúce alebo nadbytočné kroky). Na základe procesných máp je tiež možné pripraviť simuláciu procesu alebo kalkulovať náklady založené na činnostiach (ABC – Activity Based Costing). Kľúčové body procesného mapovania sú (Fiala & Ministr, 2003):

- grafické znázornenie prvkov (objekty, informácie) a činností (manuálne alebo automatizované) – účelom je správne a prehľadné znázornenie,
- z procesnej mapy musí byť zreteľné, aké činnosti má systém vykonávať na základe toho, ako je systém navrhnutý,
- procesná mapa by mala byť konzistentná a hierarchicky usporiadaná – hlavné činnosti na najvyššej úrovni a detaily na nižších úrovniach,
- zaznamenávanie všetkých rozhodnutí a pravidelných hodnotení vývoja procesnej mapy.

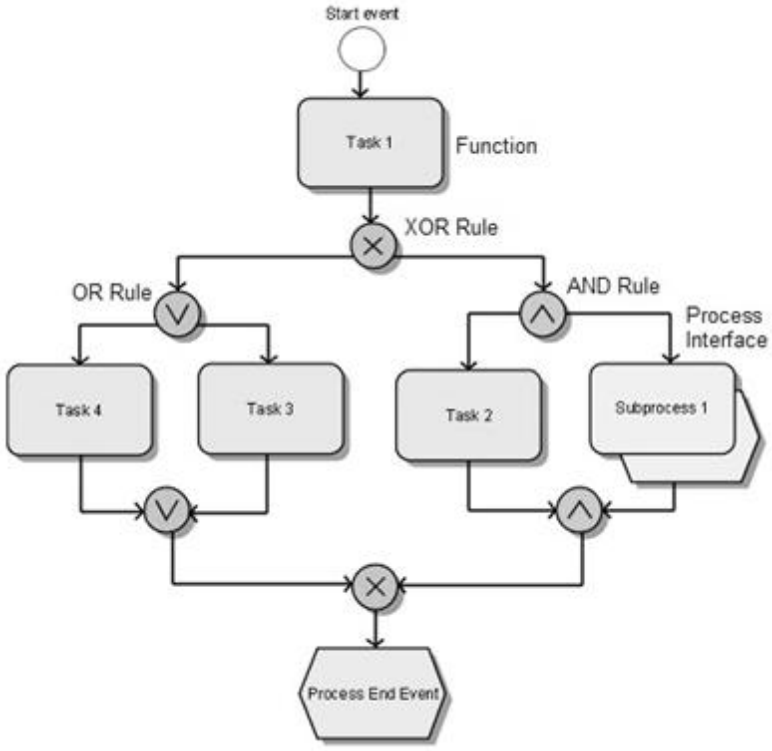
Pri nových alebo reengineeringových procesoch sa v procesnej mape špecifikujú potreby, požiadavky a funkcie procesu tak, aby čo najlepšie uspokojoval potreby zákazníkov.

Aby mohli byť procesy správne pochopené, a tiež prehľadné, je vhodné na ich znázornenie použiť diagramový model. Jedná sa o univerzálny prostriedok, ktorému rozumie analytik, a taktiež ho je schopný pochopiť aj manažér (zákazník). Toto je dôležité pre podrobnú analýzu zákazníkových potrieb, pri ktorej je komunikácia so zákazníkom veľmi dôležitá.

Diagramový model modelujúci procesy (obr.1) sa môže nazvať viacerými spôsobmi. V praxi je možné sa bežne stretnúť s názvami ako sú: procesná mapa, procesný diagram, workflow diagram, diagram dátových tokov alebo tiež diagram aktivít (IBM, 2014).

Diagramy sa používajú hlavne pri mode-lovaní, teda pri tvorbe definície procesu a následne aj pri simulácii namodelovaných procesov ako aj pri ich následnej optimalizácii.

Obr. 1: Procesný model (diagram)



Zdroj: www.ibm.com

2. REPREZENTÁCIA SIEŤOVÝCH GRAFOV

Najstaršou a najjednoduchšou formou reprezentácie procesov je reprezentácia pomocou sieťových grafov, ktoré boli vyvinuté v rokoch 1950-60. Je to grafická reprezentácia činností, z ktorých sa proces skladá. Sieťový graf reprezentuje činnosti a prechody (toky) medzi jednotlivými činnosťami. Jedná sa o orientované grafy, aby bol smer toku informácií jednoznačný, a taktiež sa nezvyknú používať cykly, ktoré veľmi sťažujú následnú simuláciu nad namodelovaným procesom. Dobrým zvykom je tiež zložité a rozvetvené procesy dekomponovať na podprocesy (zachováva sa konzistencia), čím sa následné diagramy stávajú viac čitateľnými.

Aby sa o nejakom sieťovom grafe dalo povedať, že modeluje procesy, je potrebné, aby spĺňal niekoľko formálnych náležitostí. Každá činnosť je označená pomocou identifikátora, čo býva obvykle nejaké písmeno alebo číselný kód, a taktiež sa činnosť označuje ďalšími hodnotami, ktorými môžu byť napríklad čas, cena, pravdepodobnostná funkcia alebo nejaké ďalšie potrebné informácie. Pre lepšiu prehľadnosť sa v procesnom diagrame obvykle nachádza jedna štartovacia a jedna koncová udalosť.

V prípade, že tomu tak nie je, je vhodné ich doplniť a všetky začiatočné činnosti začať z jednej počiatočnej činnosti a všetky koncové činnosti zviesť do jednej koncovej. Tiež je dobrým zvykom, že čas na procesných diagramoch beží zľava doprava, ale tiež sa dá naraziť na diagram, v ktorom bude čas plynúť zhora nadol.

Sieťové grafy, ktoré sú vhodné na modelovanie procesov (formát AON) je možné reprezentovať na základe teórie grafov pomocou grafu

$$G = (V, H) \quad (1)$$

kde V je množina uzlov (činností) a H množina hrán (vzťahov medzi činnosťami). Ďalej môžeme definovať funkciu $t(h)$ – čas trvania jednej činnosti (časový úsek – hrana).

Cieľom je nájsť pre danú skupinu bodov $U \subset V$ taký strom (minimálny strom) $S = (V', H'); V' \subset V; H' \subset H$, ktorý spája množinu bodov U a jeho celkový čas (doba trvania celého procesu)

$$T(S) = \sum_{h \in H} t(h) \quad (2)$$

bude minimálny. Vrcholy tohto stromu tvorí cieľová množina *U* (základné činnosti) a množina Steinerových vrcholov (vedľajšie/podporné činnosti). Nájdenie Steinerových vrcholov je však matematicky veľmi zložité a čas riešenia závisí exponenciálne na počte uzlov v grafe *G*. Vzniknutý problém sa dá riešiť transformáciou siete (grafu) na taký podgraf *G'* (minimálna kostra grafu), ktorý obsahuje len uzly cieľovej množiny. Nájdenie hrán s minimálnym časom spájajúcich cieľovú množinu sa môže realizovať *Floydovým algoritmom*, opakovaným použitím *Dijkstrovho algoritmu* alebo *Primovho algoritmu,* ktorý zistí najkratšie spojenia z jedného (zdrojového) uzla do všetkých ostatných uzlov v sieťovom grafe. Potom už možno vykonať spomenutú redukciu na podgraf pozostávajúci z uzlov cieľovej množiny (Filanová, 2006).

3. REPREZENTÁCIA STROMOV

V našom prípade je hlavným cieľom nájsť strom s minimálnou, alebo aspoň čo najnižšou dobou trvania celého procesu. Vzhľadom na to sa stáva kľúčovou otázkou vhodná reprezentácia takýchto stromov. Požiadavky kladené na reprezentáciu stromov možno z hľadiska dôležitosti zhrnúť do nasledujúcich bodov:

1. Možnosť opísať ľubovoľný strom.
2. Opísanie každého stromu musí mať rovnaký počet bitov.
3. Malou zmenu údajov by sa mala docieliť malá zmena vhodnosti.
4. Možnosť popísať len stromy, čím by sa zmenšil počet stupňov voľnosti.
5. Možnosť jednoduchého prepočtu na štandardné formy reprezentácie vhodné na výpočet vhodnosti.

Najpoužívanejšie formy reprezentácie stromov sú nasledujúce:

- **Matica spojení** - štvorcová matica s počtom riadkov a stĺpcov rovným počtu vrcholov v grafe, pričom ak existuje v grafe hrana z uzla *i* do *j*, tak na *i*-tom riadku v *j*-tom stĺpci je 1, ak neexistuje, potom je tam 0. Nespĺňa podmienku 4 a čiastočne 5 a navyše pri rozsiahlych grafoch vyžaduje veľké množstvo pamäte.
- **Charakteristický vektor** - binárny vektor s dimenziou rovnou počtu hrán v grafe, pričom *i*-ty prvok vektora určuje príslušnosť *i*-tej hrany ku grafu. Splnené požiadavky sú rovnaké ako při matici spojení, výhodou sú však menšie pamäťové nároky.
- **Predchodcovia** - vektor s počtom prvkov rovným počtu vrcholov, pričom každý prvok musí byť schopný identifikovať ľubovoľný vrchol. Vychádza z faktu, že každý strom má svoj koreňový vrchol, z ktorého vedú cesty do všetkých ostatných vrcholov. Preto sa pre každý vrchol definuje jeho predchodca, cez ktorého sa dá dostať do koreňového vrcholu.
- **Prüfer čísla** - číslo s počtom prvkov o 2 menším ako počet vrcholov, kde každý prvok nadobúda hodnotu maximálne rovnú počtu vrcholov v grafe. Opisujú len stromy (požiadavka 4), na druhej strane však malá zmena tohto čísla spôsobí veľkú zmenu vhodnosti stromu (požiadavka 3).
- **Steinerove vrcholy** - vrcholy Steinerovho stromu okrem vrcholov, ktoré sa majú prepojiť (účastníci). Nájdenie Steinerovho stromu sa po nájdení Steinerových vrcholov stáva jednoduchou záležitosťou, keď stačí zistiť *minimálnu kostru grafu*. Tento spôsob sa javí ako najvýhodnejší, najmä čo sa týka malého počtu stupňov voľnosti.

Z hľadiska využitia evolučných algoritmov v optimalizácii procesných máp je najdôležitejšie splnenie prvých troch podmienok. Pri rozsiahlom grafe (sieťový model s veľkým množstvom činností) sa z hľadiska skrátenia doby výpočtu stáva prioritnou najmä štvrtá a v menšej miere aj piata podmienka. Žiaľ všetky stanovené podmienky úplne nespĺňa žiadna z uvedených možností reprezentácie (Filanová, 2006).

4. EVOLUČNÉ STOCHASTICKÉ OPTIMALIZAČNÉ ALGORITMY

Evolučné stochastické optimalizačné algoritmy predstavujú množinu algoritmov, ktoré využívajú evolučné procesy na riešenie úloh, hľadanie a optimalizáciu v zložitých systémoch. V klasických deterministických algoritmoch je možné priblíženie sa ku globálnemu extrému

realizáciou algoritmu viackrát s rôznymi počiatočnými podmienkami. Náhodne sa vyberie počiatočná pozícia algoritmu a ďalší postup algoritmu je deterministický. Na rozdiel od tohto postupu sú evolučné algoritmy stochastické (náhodné) počas celého priebehu výpočtu a globálny extrém nájdu takmer vždy, ale v nekonečnom čase.

Jednotlivé evolučné algoritmy sa líšia medzi sebou dobou výpočtu a schopnosťou dosiahnuť pri prehľadávaní priestoru riešení globálny extrém. Všeobecne platí, že čím je väčšia doba výpočtu, tým dôkladnejšie je preskúmaný priestor riešení. Voľba správneho algoritmu je teda kompromisom medzi dobou výpočtu a správnosťou riešenia. Medzi najznámejšie a najpoužívanejšie evolučné algoritmy patria:

- horolezecký algoritmus (Hill Climbing),
- zakázané prehľadávanie (Tabu Search),
- simulované žíhanie (Simulated Annealing),
- evolučné stratégie (Evolution Strategy),
- genetické algoritmy.

Pri všetkých algoritmoch je veľmi dôležité nastavenie jednotlivých parametrov algoritmu. Ich chybné nastavenie sa môže prejaviť tým, že algoritmus nedosiahne globálny extrém. Všetky sa vyznačujú dlhým časovým prechodom od takmer optimálnych riešení k optimálnemu riešeniu. Je preto potrebné určiť interval, v ktorom je už možné považovať takmer optimálne riešenie za optimálne, aby sa zbytočne nepredlžovala doba výpočtu. Prijateľnejší priebeh výpočtu možno dosiahnuť aj vhodnou kombináciou uvedených algoritmov, keď sa na čiastkové výsledky jedného algoritmu môže aplikovať iný algoritmus, alebo na čiastkové výsledky uvedených stochastických algoritmov sa aplikuje niektorá z deterministických metód.

5. NÁVRH OPTIMALIZÁCIE SIEŤOVÝCH GRAFOV

Náš návrh optimalizácie procesných máp pomocou sieťových grafov s využitím evolučných optimalizačných algoritmov si ukážeme na nasledujúcom príklade (Filanová, 2012).

Pri optimalizácii sieťových grafov využijeme reprezentáciu pomocou Steinerových vrcholov. Spomínaná reprezentácia je v súčasnosti najlepšou z hľadiska dimenzie prehľadávaného priestoru (zložitosť *2^(N-m)*, kde *N* je počet uzlov siete a *m* je počet prepájaných uzlov). Takisto platí jedna z dôležitých podmienok pre použitie v evolučných algoritmoch, že malou zmenou vektoru riešenia dosiahneme malú zmenu vhodnosti riešenia.

Pretože reprezentácia Steinerovými vrcholmi poskytuje iba samotnú množinu uzlov, ktoré by mohli tvoriť strom (okrem prepájaných uzlov), jej nevýhodou sa stáva o niečo dlhší a zložitejší prepočet vektoru riešenia na štandardnú formu reprezentácie (pomocou matice spojení), z ktorej možno priamo vypočítať vhodnosť riešenia (súčet časových úsekov). Je potrebné pritom použiť algoritmus na transformáciu siete iba na podmnožinu prepájaných vrcholov a vrcholov obsiahnutých vo vektore nájdeného riešenia evolučným algoritmom. Na takto transformovanú sieť už možno aplikovať algoritmus na hľadanie minimálneho stromu. Při implementácii bude tento problém efektívne vyriešený pomocou optimalizovaného Primovho algoritmu, aby sa v čo najväčšej miere minimalizovala celková doba výpočtu evolučného algoritmu. Z dôvodu diskrétnej formy riešenia zaznamenanej binárnym číslom je potrebné definovať operátor mutácie ako pre všetky bity konštantnú pravdepodobnosť zmeny (invertovania) jednotlivých bitov riešenia. Pri voľbe jedného z možných typov kríženia v evolučných algoritmoch bude použité diskrétne kríženie dvoch rodičov, keď sa skopíruje príslušná časť z jedného rodiča na základe zvoleného počtu náhodne generovaných bodov kríženia.

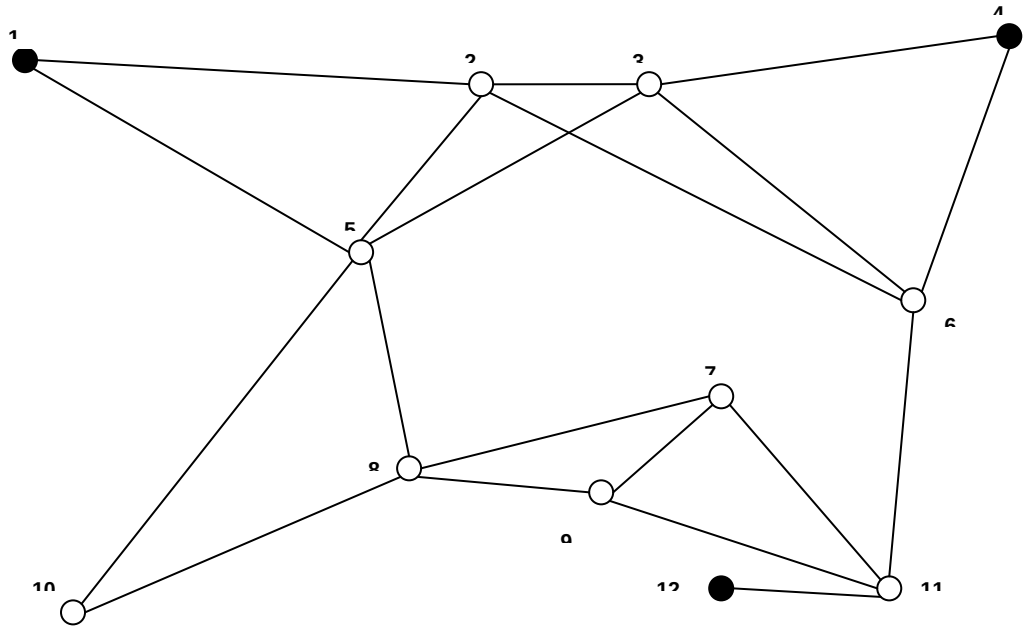
Grafický model procesnej mapy bude reprezentovaný podobne, ako pri použití matice spojení, len s tým rozdielom, že ak existuje v sieti spojenie medzi uzlom *i* a *j*, tak na *i*-tom riadku a *j*-tom stĺpci matice je miesto logickej 1 časove ohodnotenie hrany (neexistujúce spojenie je ohodnotené nulou). Predpokladom pre správne fungovanie implementovaných algoritmov je symetrickosť matice. Príklad matice spojení grafického modelu procesnej mapy je uvedený v tabuľke 1.

Tab. 1: Príklad reprezentácie procesného modelu pomocou matice spojení

uzol	1	2	3	4	5	6	7	8	9	10	11	12
1	**x**	57	0	0	48	0	0	0	0	0	0	0
2	57	**x**	19	0	25	61	0	0	0	0	0	0
3	0	19	**x**	45	41	42	0	0	0	0	0	0
4	0	0	45	**x**	0	34	0	0	0	0	0	0
5	48	25	41	0	**x**	0	0	26	0	58	0	0
6	0	61	42	34	0	**x**	0	0	0	0	35	0
7	0	0	0	0	0	0	**x**	39	18	0	31	0
8	0	0	0	0	26	0	39	**x**	23	45	0	0
9	0	0	0	0	0	0	18	23	**x**	0	37	0
10	0	0	0	0	58	0	0	45	0	**x**	0	0
11	0	0	0	0	0	35	31	0	37	0	**x**	20
12	0	0	0	0	0	0	0	0	0	0	20	**x**

Zdroj: vlastné spracovanie

Obr. 1: Reprezentácia procesného modelu pomocou sieťového grafu



Zdroj: vlastné spracovanie

Sieťový graf (obr. 2) pozostáva z dvanástich uzlov a doba činnosti medzi dvomi uzlami v tomto prípade zodpovedá dĺžke ich spojnici. Do množiny základných činností boli vybraté činnosti označené číslami 1, 4, a 12.

Každé riešenie generované evolučným algoritmom je reprezentované pomocou Steinerových vrcholov, t.j. bitovým poľom s dĺžkou rovnou počtu uzlov siete okrem prepájaných uzlov, ktoré sú vždy zahrnuté vo výsledku, a preto ich netreba uchovávať vo vektore riešenia (tab. 2).

Tab. 2: Riešenie reprezentované Steinerovými vrcholmi

číslo vrcholu	2	3	5	6	7	8	9	10	11
vektor riešenia	0	1	1	1	0	0	0	0	1

Zdroj: vlastné spracovanie

Pozícia bitu v logickom poli jednoznačne reprezentuje uzol sieťového grafu a hodnota špecifikuje, či je uzol zahrnutý v danom riešení. Všetky riešenia majú preto rovnakú dĺžku a líšia sa len obsiahnutými logickými hodnotami. Na takto reprezentované riešenie možno už bez problémov aplikovať operácie mutácie a kríženia typické pre evolučné algoritmy.

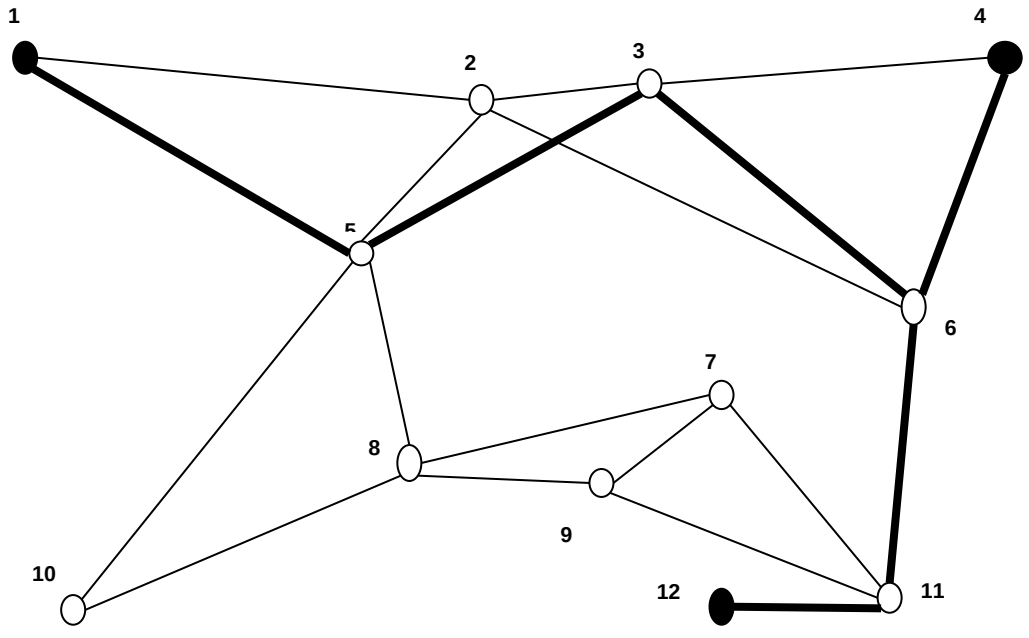
Z hľadiska nevyhnutnosti zistenia ohodnotenia jednotlivých hrán, no aj z dôvodu vhodného zobrazenia výsledného riešenia treba vykonať konverziu výsledkov do zrozumiteľnejšej formy. Preto sa z každého vektora riešenia musia zistiť bity nastavené na 1 a podľa ich pozície určiť, ktoré uzly sieťového grafu by mali byť zahrnuté v množine vrcholov hľadaného stromu. Následne sa do spomínanej množiny (Steinerove vrcholy) pridajú vybraté uzly (základné činnosti), čím získame úplnú množinu vrcholov hľadaného stromu (čísla uzlov v tabuľke 3). V ďalšom logickom kroku výpočtu nájde optimalizovaný Primov algoritmus na základe tejto množiny strom s minimálnym časom (tab. 3 a obr. 3). Optimalizácia Primovho algoritmu spočíva v časovo a pamäťovo menej náročnom vyhľadávaní v celého sieťového grafu bez jeho potreby transformácie na sieť obsahujúcu len uzly úplnej množiny vrcholov hľadaného stromu.

Tab. 3: Riešenie transformované pomocou Primovho algoritmu na maticu spojení

uzol	1	3	4	5	6	11	12
1	**x**	0	0	48	0	0	0
3	0	**x**	0	41	42	0	0
4	0	0	**x**	0	34	0	0
5	48	41	0	**x**	0	0	0
6	0	42	34	0	**x**	35	0
11	0	0	0	0	35	**x**	20
12	0	0	0	0	0	20	**x**

Zdroj: vlastné spracovanie

Obr. 2: Strom nájdený Primovým algoritmom



Zdroj: vlastné spracovanie

Ak Primov algoritmus strom nenájde, znamená to, že strom obsahujúci takúto množinu vrcholov neexistuje a treba vygenerovať nové riešenie (mutáciou resp. krížením). Tento postup sa bude opakovať až do chvíle, kým vygenerované riešenie nebude predstavovať strom. V tom prípade tvorí výstup Primovho algoritmu matica podľa tabuľky 3 (rovnakého typu ako je matica procesného modelu) reprezentujúca nájdený strom. Nakoniec sa už len jednoducho vypočíta výsledný čas stromu, ktorý je tvorený súčtom hodnôt jednotlivých vetiev (hrán).

Pre funkciu evolučných algoritmov je potrebné do celkovej optimalizácie implementovať základné funkčné operácie, ako:

- generovanie náhodných zmien konfigurácií systému = operátor mutácie,
- vzájomné kríženie dvoch riešení = operátor kríženia,
- minimalizovaná funkcia ohodnotenia = celkový čas stromu (celková doba procesu).

Porovnaním výsledkov dosiahnutých pomocou vybratých deterministických algoritmov s výsledkami dosiahnutými pomocou evolučných algoritmov sme zistili, že súčty metriky hrán výsledných viacbodových spojení pomocou evolučných algoritmov dosahujú nižšie hodnoty ako je to v prípade využitia Primovho a Greedy algoritmov. Je to dôsledok toho, že evolučné algoritmy zahŕňajú do výpočtu všetky uzly sieťového grafu (podobne ako pri využití neuronových sietí (Filanová, 2014)), pričom deterministické algoritmy pracujú len s uzlami cieľovej množiny.

ZÁVER

V článku je prezentovaný jeden z výstupov projektu s názvom Nové trendy v modelovaní podnikových procesov a optimalizácia procesných máp v podnikoch. Projekt sa zaoberá využitím informačno-komunikačných technológií v riadení podnikových procesov. Cieľom vedeckého projektu je aplikácia známych metód matematického a experimentálneho modelovania a vývoj nových metód analýzy vnútropodnikových procesov. Hlavným cieľom projektu je využitie efektívnych nástrojov na riešenie procesného modelovania a na základe optimalizácie procesov zlepšiť fungovanie a napredovanie akéhokoľvek typu podniku. Na simuláciu modelovania podnikových procesov a optimalizáciu procesných máp bude využitý optimalizačný softvér, ktorého vývoj bude súčasťou projektu.

Optimalizácia procesov s využitím evolučných algoritmov môže pomôcť eliminovať nepotrebné činnosti a podprocesy, a dokáže tak zefektívniť celý proces a skrátiť celkovú dobu procesu. Ak časovú funkciu *t(h)* nahradíme inou veličinou, napríklad nákladmi na vykonanie jednej činnosti, môžeme využiť modelovanie pomocou sieťových grafov a evolučné algoritmy na optimalizáciu nákladov na celý proces.

V článku je podrobne rozobraný vlastný návrh na efektívnu optimalizáciu procesných máp pomocou evolučných stochastických algoritmov. V rámci ďalšieho výskumu bude tento návrh implementovaný do simulátora viacbodových spojení. Následne tak môžeme simulovať modelovanie a optimalizáciu konkrétnych podnikových procesov.

Táto práca bola podporená Vedeckou agentúrou VEGA prostredníctvom finančnej podpory projektu č. 1/0336/14.

LITERATURA

Fiala, J., & Ministr, J. (2003). *Průvodce analýzou a modelováním procesů.* Ostrava: Vysoká škola báňská - Technická univerzita.

Filanová, J. (2006). Optimalizácia smerovania viacbodových spojení v telekomunikačných sieťach pre potreby videokonferencie, (Dizertačná práca), Bratislava: Slovenská technická univerzita v Bratislave.

Filanová, J. (2013). Využitie evolučných stochastických algoritmov v optimalizácii procesných máp. *Vybrané problémy informačného manažmentu*. (1-15). Ekonóm.

Filanová, J. (2014). Modelovanie procesných máp pomocou neurónových siet. *Vybrané*

problémy informačného manažmentu (1-13). EKONÓM.

Kvasnička, V., Pospíchal, J., & Tiňo, P. (2000). *Evolučné algoritmy.* Bratislava: Vydavateľstvo STU.

IBM. Dostupné na: <http://www.ibm.com>.

Adresa autora:
Ing. Jana Filanová, PhD.,
Ekonomická univerzita v Bratislave.
Fakulta podnikového manažmentu
Katedra informačního manažmentu
jana.filanova@euba.s

OPTIMIZATION OF PROCESSES USING EVOLUTIONARY ALGORITHMS

Jana Filanová

Abstract: The paper deals with process management, process modeling and optimization of process maps. Modeling process maps based on network graphs is presented in the paper. Applying stochastic evolutionary algorithm in process maps optimization is described.

Key words: Business process management, Evolution stochastic algorithms, Optimization of processes

JEL Classification: C61, M21, O3