UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



Veľké dáta a DEA modely

DIPLOMOVÁ PRÁCA

2021 Bc. Martin FAKTOR

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Veľké dáta a DEA modely

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomicko-finančná matematika a modelovanie
Študijný odbor:	1114 Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	doc. RNDr. Margaréta Halická, CSc.

Bratislava 2021 **Bc. Martin FAKTOR**





Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Martin Faktor
Študijný program: ekonomicko-finančná matematika a modelovanie (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Veľké dáta a DEA modely
Big data and DEA models

Anotácia: Problematika spracovania veľkých dát sa vyvinula do samostatnej vednej disciplíny data science a terminológia z nej preniká aj do iných príbuzných oblastí, v ktorých sa spracúvajú dáta. Je tomu tak aj v prípade obálkovej analýzy dát, označovanej ako DEA, kde sa v ostatných rokoch publikovalo niekoľko článkov navrhujúcich špeciálne postupy pre riešenie DEA modelov v prípade veľkých dát. Práca bude analyzovať opodstatnenosť použitia termínu 'big data' v súvislosti s reálnymi aplikáciami DEA a efektívnosť navrhovaných postupov oproti použitiu metód vnútorného bodu.

Vedúci: doc. RNDr. Margaréta Halická, CSc.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Marek Fila, DrSc.

Dátum zadania: 16.01.2019

Dátum schválenia: 17.01.2019 prof. RNDr. Daniel Ševčovič, DrSc.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie Touto cestou by som chcel úprimne poďakovať svojej vedúcej diplomovej práce doc. RNDr. Margaréte Halickej, CSc. najmä za jej čas, ochotu, pomoc, podnetné rady a cenné pripomienky, ktoré mi venovala pri písaní tejto práce. Zároveň by som taktiež veľmi rád poďakoval svojej rodine a blízkym za ich podporu.

Abstrakt v štátnom jazyku

FAKTOR, Martin: Veľké dáta a DEA modely [Diplomová práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školiteľ: doc. RNDr. Margaréta Halická, CSc., Bratislava, 2021

V diplomovej práci sa zaoberáme špecifickými algoritmami obálkovej analýzy dát, ktoré slúžia najmä k signifikantnej redukcii výpočtového času. V prvej časti uvádzame fundamentálnu terminológiu obálkovej analýzy dát ako aj tvrdenia s príslušnými dôkazmi. Súčasťou tejto časti je taktiež aditívny model ako aj jeho modifikácia pre výpočet superefektivity. Keďže tradičný prístup výpočtu DEA modelov sa nemusí javiť ako postačujúci na riešenie úloh, za predpokladu prítomnosti dát veľkých rozmerov, obsahom druhej časti je spracovanie a porovnanie rôznych algoritmov vedúcich k zrýchleniu výpočtu a zároveň predstavenie vlastného algoritmu. V poslednej časti práce porovnávame vlastný algoritmus s ďalšími predstavenými algoritmami, naprogramovanými v softvéri MATLAB, na dátach rôznych kardinalít, rozmerov a hustôt.

Kľúčové slová: obálková analýza dát, aditívny model, lineárne programovanie, veľké dáta

Abstract

FAKTOR, Martin: Big data and DEA models [Diploma Thesis], Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics; Supervisor: doc. RNDr. Margaréta Halická, CSc., Bratislava, 2021

In this diploma thesis, we investigate specific algorithms of data envelopment analysis which allow for a significant reduction of computation time. In the first section, we introduce basic terminology of data envelopment analysis and theorems with respective proofs. Part of this section is also an additive model and its modification for calculations of super-efficiency. The traditional approach to DEA models calculations may not be sufficient for problem-solving with high-dimensional data content. Therefore, in the second section, we analyse and compare various algorithms which enable accelerated calculations and introduce a novel algorithm developed by us. In the last section, we compare our algorithm with other existing algorithms implemented in MATLAB and test on data with various sizes, densities and cardinalities.

Keywords: Data Envelopment Analysis, additive model, linear programming, big data

Obsah

Úvod **8**

1 Teoretické základy **10**

1.1 Terminológia a symbolika DEA modelov 10

1.2 Množina produkčných možností a jej aproximácie 12

1.3 Hranica množiny produkčných možností 14

1.4 Efektívny vzor 16

1.5 Aditívny model a superefektivita 19

1.5.1 Formulácia aditívneho modelu 20

1.5.2 Vlastnosti aditívneho modelu 21

1.5.3 Superefektivita pre aditívny model 24

2 Algoritmy znižujúce výpočtovú náročnosť **27**

2.1 Motivácia 27

2.2 Prehľad algoritmov 29

2.2.1 Restricted basis entry metóda 30

2.2.2 Metóda hierarchickej dekompozície 35

2.2.3 Build Hull 40

2.2.4 Návrh nového algoritmu 47

3 Numerické experimenty **53**

3.1 Numerické výsledky 53

3.2 Zhrnutie výsledkov 60

Záver **62**

Zoznam použitej literatúry **64**

Príloha A **66**

Úvod

V dnešných časoch, teda v dobe informačnej revolúcie a nebývalého technologického pokroku, zažíva problematika veľkých dát, označovaná v angličtine ako *Big data*, nebývalý rozmach. V súčasnej dobe je badať rastúci trend zhromažďovania a spracovania dát pochádzajúcich z rôznych sektorov. Taktiež je veľmi pravdepodobné, že so zhromaždenými dátami sa obchoduje ako s určitou nehmotnou komoditou. Následkom tohto obchodu je rastúci apel vlastníkov dát na ich rôznorodé analýzy.

Výsledkom analýz naprieč výrobnými ako aj nevýrobnými odvetviami môže byť napríklad zodpovedanie otázky tykajúcej sa vyhodnotenia efektívnosti. Spomenutý pojem tvorí ťažisko časti operačného výskumu nazývaného obálková analýza dát (DEA). DEA modely slúžia najmä na vyhodnocovanie efektívnosti producentov a následnej klasifikácie producentov na efektívnych resp. neefektívnych. DEA modely sa tak môžu v nemalej miere spolupodieľať na lepšom fungovaní činností v rôznych sférach. Môžeme predpokladať, že neexistuje oblasť, ktorá by sa nesnažila pri produkcii efektívnejšie zaobchádzať so svojimi zdrojmi.

Ako veľká nevýhoda použitia DEA modelov sa javí veľká časová náročnosť ich výpočtu. Tradičný spôsob výpočtu totiž môže predstavovať, napriek enormnej výpočtovej sile súčasných zariadení, problém aplikácie DEA modelov na veľké dátové sety. V posledných rokoch sa začínajú objavovať veľké dátové sety vhodné na aplikáciu zmienených DEA modelov. Možnosť aplikácie však naráža na vyššie spomenutý problém časovej náročnosti výpočtu a s ním súvisiacej neschopnosti promptnej klasifikácie pozorovaných útvarov. Tento problém podnietil vznik špeciálnych algoritmov [1], [3], [8], vedúcich k nemalým časovým úsporám.

Cieľom tejto diplomovej práce je spracovať literatúru venujúcu sa zrýchľujúcim algoritmom obálkovej analýzy dát a na základe princípov v nich obsiahnutých navrhnúť vlastný prístup vedúci k časovým úsporám.

V prvej časti práce sa budeme najskôr venovať teoretickým základom DEA modelov, zavedeniu symboliky a terminológie, ktoré budú tvoriť nevyhnutnú súčasť pre následne predstavené úvahy. Okrem fundamentálnych poznatkov popíšeme aditívny model, pričom poukážeme na niektoré z jeho vlastností. Dôležitou súčasťou tejto kapitoly bude aj modifikácia aditívneho modelu vedúca k výpočtu superefektivity.

Druhá kapitola bude venovaná špeciálnym algoritmom, ktoré majú za cieľ signifikantným spôsobom znížiť celkový výpočtový čas. Zameriame sa primárne na dôkladné spracovanie literatúry, ktorá bola v posledných rokoch publikovaná a pojednáva o vylepšených algoritmoch, ktoré sa javia ako veľmi vhodné pri zaobchádzaní s dátovými sadami veľkých rozmerov. V tejto kapitole tiež predstavíme náš vlastný algoritmus, ktorý bude využívať modifikáciu aditívneho modelu pre výpočet superefektivity, ako aj určité princípy vychádzajúce z dokázaných tvrdení v rámci prvej kapitoly.

V poslednej časti práce sa budeme venovať testovaniu algoritmov predstavených v druhej kapitole ako aj navrhnutého algoritmu na veľkých dátových sadách rôznych rozmerov a hustôt, pričom sa budeme snažiť taktiež kriticky zhodnotiť potreby vývoja ďalších algoritmov vedúcich k zrýchleniu výpočtu v DEA problematike v budúcnosti.

1 Teoretické základy

Naša práca má za cieľ objasniť rôzne druhy algoritmov, slúžiace k zrýchleniu evaluácie modelov obálkovej analýzy dát v prípade prítomnosti dátových sád veľkých rozmerov. Výsledné zhodnotenie efektívnosti navrhnutých algoritmov oproti bežne používaným postupom je taktiež náplňou tejto práce. Najprv je však potrebné predstaviť a uviesť základnú symboliku, terminológiu a definície potrebné k neskoršiemu použitiu v teórii obálkovej analýzy dát. Táto časť diplomovej práce bude slúžiť taktiež k predstaveniu základného aditívneho modelu, ako aj jeho určitej modifikácie, ktorá bude neskôr využitá pri konštrukcii nami navrhnutého algoritmu vedúceho k zníženiu celkového výpočtového času. Pri spracovaní tejto kapitoly sme čerpali prevažne z učebného textu [14], ale taktiež sme využívali poznatky z [15], [19].

1.1 Terminológia a symbolika DEA modelov

Ústredné pojmy efektívnosť a efektivita sa viažu v DEA modeloch na určitý spôsob vykonávania nejakej činnosti. Túto činnosť budeme, v súlade s ekonomickou teóriou, nazývať *technológiou* alebo taktiež produkčnou činnosťou prípadne aj produkciou. Znamená to, že technológiu môžeme chápať ako istý proces produkčnej transformácie, ktorá je charakterizovaná spotrebovaním vstupov na produkciu výstupov. Vo väčšine literatúry, týkajúcej sa obálkovej analýze dát platí úzus, že technológia spotrebováva m vstupov, ktoré sú v literatúre zvyčajne označované ako $I_1, \ldots, I_m$ (z angl. *Input*), na produkciu s výstupov, ktoré sú označované ako $O_1, \ldots, O_s$ (z angl. *Output*). Tento transformačný proces vstupov na výstupy pomocou technológie T, môžeme symbolicky znázorniť nižšie priloženým obrázkom.



Obr. 1: Schematické znázornenie Technológie T.

Zdroj: [14]

Ako modelový príklad technológie T si môžeme predstaviť cestovnú kanceláriu, pričom jej vstupy môžu tvoriť napríklad výplaty zamestnancov a náklady na chod kancelárie, a naopak jej výstupy môžu byť tvorené celkovou cenou predaných zájazdov. Znamená to, že v nami uvedenom konkrétnom príklade, technológia T spotrebováva 2 vstupy na produkciu 1 výstupu. Vo všeobecnosti vstupom technológie T zodpovedá m-rozmerný vektor $x \in \mathbb{R}^m$ a výstupom zodpovedá s-rozmerný vektor $y \in \mathbb{R}^s$. Môžeme teda hovoriť, že pre dvojicu (x, y) platí, že z množstiev x vstupov sa dá, vzhľadom k technológií T, vyrobiť množstvo y výstupov. Na základe predošlého teda hovoríme, že dvojica (x, y) je pre technológiu T prípustná.

Konkrétne dvojice (x, y) sa v terminológii DEA modelov viažu k dôležitému pojmu, ktorým je *rozhodovacia jednotka DMU* (z angl. *Decision Making Unit*). Vo väčšine DEA literatúry je obvykle zaužívané, že indexácii jednotlivých rozhodovacích jednotiek DMU je priradené písmeno j a počet všetkých pozorovaných útvarov označujeme n. Teda j-ty útvar budeme označovať DMU_j, pričom $j \in \{1, \ldots, n\}$, a všetky rozhodovacie jednotky dokopy tvoria množinu n pozorovaní.

Je prirodzené vyžadovať homogenitu danej skupiny útvarov. Pod požiadavkou homogenity skupiny producentov rozumieme, že všetky pozorované útvary podliehajú rovnakej produkčnej činnosti. Teda má zmysel porovnávať napríklad cestovné kancelárie medzi sebou, ako v ilustračnom príklade uvedenom vyššie, ale nemá zmysel porovnávať cestovné kancelárie napríklad s pobočkami bánk prípadne elektrárňami a podobne. Homogenita producentov teda zabezpečuje vyhodnocovanie relatívnych efektivít, v rámci rovnakej produkčnej činnosti. Ak chceme upriamiť pozornosť na konkrétnu rozhodovaciu jednotku, teda zistiť jej relatívnu efektivitu v rámci určitej skupiny homogénnych producentov, je potrebné jej priradiť odlišný index ako zvyšku pozorovaných útvarov. Z dôvodu potreby rozlíšenia teda používame označenie DMU_o pre $o \in \{1, \ldots, n\}$.

Existujú rôzne modifikácie DEA modelov, z ktorých pramenia odlišné predpoklady na vektor vstupov resp. vektor výstupov. Táto práca však bude uvažovať len taký prípad, že pozorované subjekty spotrebovávajú iba nezáporné množstvá jednotlivých vstupov a produkujú len nezáporné množstvá výstupov. Zároveň budeme predpokladať, že množina vektorov všetkých vstupov ,resp. množina vektorov všetkých výstupov neobsahuje nulový vektor. To znamená, že požadujeme aby každý pozorovaný útvar mal aspoň jednu zložku

vektora vstupov ,resp. výstupov kladnú. Pre takto zvolené vektory platí, že $x_j \in \mathbb{R}_+^m$ resp. $y_j \in \mathbb{R}_+^s$, pričom $x_j \neq 0$ a $y_j \neq 0$. Vektorový zápis vstupov ,resp. výstupov vyzerá nasledovne:

$$x_j = \begin{pmatrix} x_{1j} \\ \vdots \\ x_{ij} \\ \vdots \\ x_{mj} \end{pmatrix}, \quad y_j = \begin{pmatrix} y_{1j} \\ \vdots \\ y_{rj} \\ \vdots \\ y_{sj} \end{pmatrix}, \quad \forall j = 1, \ldots, n.$$

Formulácie DEA modelov sa častokrát vyskytujú aj v tzv. maticovom zápise, pričom tieto matice budú tvoriť taktiež vstupné údaje v praktickej časti tejto diplomovej práce a z týchto dôvodov ich nižšie uvádzame. Množine vektorov vstupov a vektorov výstupov všetkých pozorovaných útvarov DMU_j pre $j \in \{1, \ldots, n\}$ zodpovedajú matica vstupov X rozmeru $m \times n$,resp. matica výstupov Y rozmeru $s \times n$. J-tym stĺpcom matíc X, Y zodpovedá vektor vstupov ,resp. vektor výstupov j-teho pozorovaného útvaru DMU_j pre $j \in \{1, \ldots, n\}$. Maticový zápis vstupov ,resp. výstupov všetkých pozorovaných útvarov vyzerá nasledovne:

$$X = \begin{pmatrix} x_{11} & \ldots & x_{1j} & \ldots & x_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i1} & \ldots & x_{ij} & \ldots & x_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{m1} & \ldots & x_{mj} & \ldots & x_{mn} \end{pmatrix}, \quad Y = \begin{pmatrix} y_{11} & \ldots & y_{1j} & \ldots & y_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{r1} & \ldots & y_{rj} & \ldots & y_{rn} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{s1} & \ldots & y_{sj} & \ldots & y_{sn} \end{pmatrix}.$$

1.2 Množina produkčných možností a jej aproximácie

Ďalším dôležitým pojmom vystupujúcim v terminológii obálkovej analýze dát je pojem pochádzajúci z mikroekonomickej teórie, ktorým je *množina produkčných možností* (angl. *Production possibility set*) ozn. G. Množina produkčných možností spravidla nie je známa, avšak sú odvodené a popísané určité jej vlastnosti, ktoré sú využívané pri konštrukcii jej aproximácie. Aproximáciu množiny produkčných možností G budeme označovať symbolom M. Existuje mnoho rôznych podôb množiny M, ktoré sa navzájom líšia. Rozdiely

vo výsledných konštrukciách spôsobuje napríklad výber odlišných výnosov z rozsahu. V ďalšom uvedieme dve základné aproximácie množiny G. Konkrétne sa bude jednať o aproximačnú množinu produkčných možností vzhľadom ku konštantným výnosom z rozsahu CRS (z angl. *Constant return to scale*) a aproximačnú množinu produkčných možnosti vzhľadom k variabilným výnosom z rozsahu VRS (z angl. *Variable return to scale*). Ešte predtým, než prejdeme k samotným formuláciám axióm, zavedieme množinu obsahujúcu všetky pozorované útvary [14]:

$$Z := \{(x_j, y_j) \in \mathbb{R}_+^{m+s} | j \in \{1, \ldots, n\}\}. \tag{1}$$

Formulácie aproximačných množín M, sa opierajú o nasledovné axiómy [14]:

(A1) $Z \subset M$. Čo znamená, že všetky pozorované dvojice patria množine M.

(A2) M je konvexná, čiže každá konvexná kombinácia dvoch bodov patriacich do M patrí tiež do množiny M. To znamená že platí, že ak pozorovaným producentom DMU_A a DMU_B prislúchajú dvojice vektorov (x_A, y_A) ,resp. $(x_B, y_B) \in M$, potom pre ľubovoľné $\lambda \in [0, 1]$ musí nutne platiť, že aj dvojica $(\lambda x_A + (1 - \lambda)x_B, \lambda y_A + (1 - \lambda)y_B) \in M$.

(A3) Ak producentovi DMU_A prislúcha dvojica $(x_A, y_A) \in M$, potom do množiny M patria aj všetky body $(x, y) \in \mathbb{R}^{m+s}$, pre ktoré platí $x_A \leq x \wedge y \leq y_A$. Táto vlastnosť je označovaná ako *voľná disponovanosť* (z angl. *free disposability*). Voľná disponovanosť hovorí, že je možné plytvať vstupmi, čomu zodpovedá nerovnosť $x_A \leq x$, a zároveň produkovať nedostatočné množstvá výstupov, čomu zodpovedá nerovnosť $y \leq y_A$.

Najmenšiu množinu spĺňajúcu vyššie uvedené 3 axiómy budeme chápať ako aproximačnú množinu produkčných možností vzhľadom k variabilným výnosom z rozsahu a budeme ju označovať ako M_{VRS}. Pre potreby konštrukcie aproximačnej množiny vzhľadom ku konštantným výnosom z rozsahu je potrebné uviesť ešte jednu axiómu. Táto axióma je typická práve pre daný druh výnosov z rozsahu a hovorí, že ľubovoľný kladný násobok dvojice $(x, y) \in M$ taktiež patrí M. Alebo inak povedané, s ktorýmkoľvek bodom množiny M patrí do tejto množiny celá polpriamka vychádzajúca z počiatku súradnicovej sústavy a prechádzajúca daným bodom. To sa ekonomicky dá interpretovať aj takým spôsobom,

že každý producent vzhľadom k svojím vstupom a výstupom môže c-násobne zmeniť svoju produkciu pri c-násobnom zmenení svojej spotreby. Spomenutá štvrtá axióma môže byť formulovaná nasledovným spôsobom [14]:

(A4) Ak producentovi DMU_A prislúcha dvojica $(x_A, y_A) \in M$, potom platí, že $(cx_A, cy_A) \in M$ pre $\forall c > 0$.

Najmenšiu množinu spĺňajúcu vyššie uvedené štyri axiómy budeme chápať ako aproximačnú množinu produkčných možností vzhľadom ku konštantným výnosom z rozsahu a budeme ju označovať ako M_{CRS}. Axiómy (A1)-(A4) sú postačujúce k analytickým zápisom aproximačných množín M_{CRS} a M_{VRS}. Množina M_{CRS} sa dá na ich základe analyticky formulovať v nasledovnom tvare [14]:

$$M_{CRS} = \{(x, y) \in \mathbb{R}^{m+s} \mid X\lambda \leq x, \quad Y\lambda \geq y, \quad \lambda \geq 0\}. \tag{2}$$

Analogicky môžeme naformulovať aj množinu M_{VRS}, ktorej zápis vychádza z prvých troch axióm. Táto zmena sa v analytickom zápise prejaví pridaním podmienky $\sum_{j=1}^{n} \lambda_j = 1$. Berúc do úvahy túto podmienku, množinu M_{VRS} môžeme zapísať ako [14]:

$$M_{VRS} = \{(x, y) \in \mathbb{R}^{m+s} \mid X\lambda \leq x, \quad Y\lambda \geq y, \quad \lambda \geq 0, \quad \sum_{j=1}^{n} \lambda_j = 1\}. \tag{3}$$

Poznamenajme, že na základe analytických zápisov jednotlivých aproximačných množín je zrejmé, že platí $M_{VRS} \subseteq M_{CRS}$. Uvedené množiny M_{VRS} ,resp. M_{CRS} sú v DEA literatúre označované ako obálka dát daných producentov, pričom tento pojem dal názov aj samotnej problematike. Obálka dáva veľmi priamočiaru interpretáciu celej podstaty DEA, čo uvidíme neskôr.

1.3 Hranica množiny produkčných možností

Množina produkčných možností M, kde M, tak ako sme zaviedli označuje M_{CRS} alebo M_{VRS}, tvorí neohraničený konvexný polyéder. Tento polyéder je možné jednoznačne vyjadriť konečným prienikom polpriestorov, pričom každý z polpriestorov je jednoznačne určený nadrovinou. V nasledovnom uvedieme definíciu opornej nadroviny množiny produkčných možností M, pričom pri jej formulácii sme vychádzali z publikácie [15].

Definícia 1.1. *Nech* $(u, v) \in \mathbb{R}_+^{m+s}$, $\sigma \in \mathbb{R}$ *a nech* $(\hat{x}, \hat{y}) \in M$. *Potom nadrovina* $\mathbb{H}(u, v, \sigma) = \{(x, y) \in \mathbb{R}_+^{m+s} | y^T u - x^T v + \sigma = 0\}$ *sa nazýva opornou nadrovinou množiny* M *v bode* $(\hat{x}, \hat{y})$, *ak platí:*

(i) $y^T u - x^T v + \sigma \leq 0$ *pre všetky* $(x, y) \in M$ *a*

(ii) $\hat{y}^T u - \hat{x}^T v + \sigma = 0$.

Obzvlášť dôležitým pojmom vystupujúcim v obálkovej analýze dát je pojem *hranice*. Spoločnou črtou efektívnych útvarov je ich poloha, ktorá bude súvisieť s časťou hranice konvexného polyédra. Z tohto dôvodu uvádzame nasledovnú definíciu:

Definícia 1.2. *Množinu bodov* $(x, y) \in M$, *pre ktoré platí, že v ich ľubovoľne malom okolí existuje bod* (x_N, y_N), *ktorý nepatrí množine* M, *nazývame hranicou množiny* M *a označujeme* ∂M.

Ako sme na začiatku kapitoly uviedli, ťažiskovými pojmami obálkovej analýzy dát sú efektivita a efektívnosť. Ak ich však chceme presnejšie vymedziť, je potrebné zaviesť pojem dominujúceho útvaru.

Definícia 1.3. *Nech* $(x_o, y_o) \in M$. *Pod dominujúcim útvarom* (x, y), *ktorý dominuje útvaru* (x_o, y_o) *v množine* M, *rozumieme ľubovoľný útvar* $(x, y) \in M$ *taký, že platí* $x \leq x_o$ *a* $y \geq y_o$.

Využitím predošlej definície o dominujúcom útvare môžeme uviesť definíciu efektívneho útvaru. Nižšie uvádzame túto definíciu, ktorá sa taktiež označuje aj ako Pareto-Koopmansova definícia efektívnosti, pričom pri formulácii čerpáme z článku [15].

Definícia 1.4. *Útvar* $(x_o, y_o) \in M$ *nazývame efektívnym v* M *vtedy a len vtedy, keď neexistuje také* $(x, y) \in M$, *že platí* $x \leq x_o$ *a* $y \geq y_o$ *a zároveň* $(x_o, y_o) \neq (x, y)$.

V súlade s Pareto-Koopmansovou definíciou efektívnosti nazývame efektívnymi iba také útvary, pre ktoré sa v množine M nenachádzajú žiadne iné útvary, ktoré by umožňovali z nižšej alebo rovnakej veľkosti spotreby vstupov produkovať vyššie alebo rovnaké veľkosti výstupov. Inak povedané, v uzavretom polyedrickom kuželi s vrcholom v bode (x_o, y_o) tvorenom smerovými vektormi zodpovedajúcimi rastúcim výstupom a klesajúcim vstupom, sa nenachádza žiaden iný bod okrem (x_o, y_o). Je zrejmé, že útvary, ktoré spĺňajú

vyššie uvedenú podmienku, ležia na tzv. *hranici efektívnosti*. Táto časť hranice množiny M je ozn. H_E a jedná sa o súčasť celkovej hranice aproximačnej množiny produkčných možností ∂M. Hranicu efektívnosti množiny M môžeme analyticky podľa [14] zapísať v tvare:

$$H_E = \{(x, y) \in \partial M \mid \nexists (x^*, y^*) \in M : (x^*, y^*) \neq (x, y), x^* \leq x, y^* \geq y\} \tag{4}$$

Hranica efektívnosti H_E množiny M je tvorená časťami oporných nadrovín konvexného polyédra, ktoré sú rôznobežné so všetkými nadrovinami, pre ktoré platí $x_i = 0$ pre niektoré $i \in \{1, \ldots, m\}$ alebo $y_r = 0$ pre niektoré $r \in \{1, \ldots, s\}$. V prípade, že bod z M nie je efektívny, teda nie je súčasťou hranice efektívností H_E, nazývame ho neefektívnym. Pre zúplnenie hranice ∂M je potrebné ešte spomenúť, že okrem hranice efektívnosti je tvorená aj *hranicou pseudoefektívnosti*, ktorá je označovaná ako H_P. Táto časť hranice je tvorená takými neefektívnymi útvarmi, ktoré dosahujú maximálnu hodnotu niektorého z výstupov ,príp. minimálnu hodnotu niektorého zo vstupov v porovnaní so zvyšnými pozorovanými útvarmi, no zároveň vo zvyšných zložkách vykazujú isté rezervy. Teda jedná sa o také útvary, ktoré môžeme analyticky zapísať podľa [14] ako:

$$H_P = \{(x, y) \in \partial M \mid \exists (x^*, y^*) \in M : (x^*, y^*) \neq (x, y), x^* \leq x, y^* \geq y\} \tag{5}$$

Hranica pseudoefektívnosti H_P množiny M je teda tvorená takými časťami oporných nadrovín konvexného uzavretého polyédra, ktoré sú rovnobežné s niektorou z nadrovín spĺňajúcu $x_i = 0$ pre niektoré $i \in \{1, \ldots, m\}$ alebo $y_r = 0$ pre niektoré $r \in \{1, \ldots, s\}$. Úplna hranica aproximačnej množiny produkčných možností ∂M je tým pádom zjednotením hranice efektívnosti H_E s hranicou pseudoefektívnosti H_P, čiže platí $\partial H = H_E \cup H_P$ a zároveň $H_E \cap H_P = \emptyset$.

1.4 Efektívny vzor

Hlavná úloha DEA modelov nepozostáva iba z klasifikácie všetkých pozorovaných útvarov patriacich množine produkčných možností na dve disjunktné podmnožiny, konkrétne množinu efektívnych a množinu neefektívnych útvarov, ale taktiež z evaluácie miery neefektivity každého z neefektívnych útvarov. Túto mieru nazývame efektivitou a snahou jednotlivých modelov je definovať ju ako bezrozmernú veličinu. Táto bezrozmerná veličina

je zvyčajne ozn. E, pričom sa nachádza v polouzavretom intervale $(0, 1]$, a zároveň pre efektívny útvar platí $E = 1$.

Pre potreby určenia efektivity neefektívnych útvarov je potrebné uviesť definíciu vzoru a efektívneho vzoru pozorovaného útvaru DMU_j. Túto definíciu uvádzame nižšie, pričom čerpáme z učebného textu [14].

Definícia 1.5. *Nech $(x_o, y_o) \in M$. Pod vzorom bodu (x_o, y_o) v množine M rozumieme ľubovoľný bod $(x_v, y_v) \in M$ taký, že $x_v \leq x_o$ a $y_v \geq y_o$. Ak je vzor (x_v, y_v) efektívnym bodom v množine M, tak ho nazývame efektívnym vzorom.*

Na základe definície je zrejmé, že vzorom pozorovaného útvaru DMU_o je každý útvar nachádzajúci sa v prieniku konvexného polyedrického kužeľa s vrcholom v bode (x_o, y_o) s ramenami, ktoré zodpovedajú smerom rastúcich výstupov a klesajúcich vstupov a množiny produkčných možností. To znamená že sa jedná o množinu všetkých útvarov, ktoré dominujú pozorovanému útvaru DMU_o. Zároveň platí, že prienikom vyššie spomenutého kužeľa s hranicou efektivity, získame množinu všetkých efektívnych vzorov pozorovaného útvaru DMU_o.

Množina efektívnych vzorov vo všeobecnosti nie je jednoprvková. Rôzne formulácie DEA modelov dávajú odlišné odpovede na to, akým spôsobom získať efektívny vzor, ktorý sa následne podieľa na evaluácii relatívnej efektivity neefektívneho útvaru. Takýto efektívny vzor, ktorý determinuje mieru efektivity neefektívneho útvaru, budeme označovať pojmom *projekcia*. Dodávame však, že sa nemusí jednať o projekciu v matematickom zmysle slova, a zároveň nemusí byť ani jednoznačne definovaná.

V prípade, že uvažujeme variabilné výnosy z rozsahu, je možné formulovať tvrdenie, pomocou ktorého vieme pre niektoré útvary rozhodnúť, či ležia na hranici množiny produkčných možností bez potreby riešenia úlohy lineárneho programovania (LP). Dodávame, že uvedené tvrdenie nehovorí nič o efektivite takto získaných útvarov. S určitým dodatočným predpokladom, ktorý uvedieme neskôr, však budeme môcť o niektorých útvaroch tvrdiť, že sú efektívne. Využitie nasledovného tvrdenia bude demonštrované v neskorších častiach tejto práce.

Tvrdenie 1.6. *Nech pre útvar DMU_o, ktorému prislúcha dvojica (x_o, y_o) platí, že má maximálnu hodnotu niektorého z výstupov alebo minimálnu hodnotu niektorého zo vstupov v porovnaní so zvyšnými útvarmi, potom takýto útvar DMU_o leží na hranici M_{VRS}.*

Dôkaz. Vezmime útvar DMU_o s prislúchajúcou dvojicou (x_o, y_o). Bez ujmy na všeobecnosti predpokladajme, že tento útvar má maximálnu hodnotu r-tého výstupu, kde $r \in \{1, \ldots, s\}$, v porovnaní so zvyšnými pozorovanými útvarmi. To znamená, že platí $y_{ro} = \max_j y_{rj}$, kde $j \in \{1, \ldots, n\}$. Definujme nulový vektor v dimenzie m, teda platí, že $v = (0, \ldots, 0) \in \mathbb{R}^m$. Ďalej definujme jednotkový vektor u dimenzie s, ktorý má jednotku na r-tej pozícii, teda platí, že $u = (0, \ldots, 1, \ldots, 0) \in \mathbb{R}^s$. Skonštruujme nadrovinu $\mathbb{H}(u, v, \sigma)$, ktorá je daná rovnicou $y^T u - x^T v + \sigma = 0$, pričom $\sigma = -y_{ro} = -\max_j y_{rj}$. Je zrejmé, že vektor $(u, -v)$ je normálový vektor danej roviny. Môžeme si uvedomiť, že ak zvolíme $(x, y) = (x_o, y_o)$, tak útvar DMU_o spĺňa rovnosť. To však znamená, že útvar DMU_o leží na takto skonštruovanej nadrovine. Môžeme si ďalej uvedomiť, že pre ľubovoľný útvar (x, y) patriaci množine M_{VRS} platí, že $y^T u - x^T v - y_{ro} \leq 0$, keďže hodnota y_{ro} je maximálna. Je teda zrejmé, že nami skonštruovaná nadrovina spĺňa definíciu opornej nadroviny, keďže v nej leží útvar DMU_o a zároveň sme ukázali, že všetky útvary množiny M_{VRS} ležia v jednom z polpriestorov. Z toho, že sme ukázali, že útvar DMU_o patrí opornej nadrovine konvexného polyédra vyplýva, že sa nachádza na hranici ∂M_{VRS}, čo bolo potrebné dokázať. Analogicky by sme postupovali v prípade minimálnej hodnoty niektorého z m vstupov. S rozdielom, že ak útvar DMU_o s prislúchajúcou dvojicou (x_o, y_o) by mal minimálnu hodnotu i-teho vstupu, spomedzi všetkých pozorovaných útvarov, teda by platilo $x_{io} = \min_j x_{ij}$, kde $j \in \{1, \ldots, n\}$, zvolili by sme $\sigma = x_{io}$. □

Množinu produkčných možností M jednoznačne určuje sada n generujúcich útvarov daných množinou Z a axiómy uvedené na začiatku kapitoly. Vyberme ľubovoľnú neprázdnu podmnožinu $D \subseteq Z$, pričom takto zvolenú podmnožinu budeme nazývať *blokom*. Pre daný blok môžeme, na základe axióm, vytvoriť množinu produkčných možností generovanú blokom D, ktorú budeme označovať M^D, ktorá odpovedá rovnakému typu výnosov z rozsahu ako množina M, generovaná množinou Z. Pre ľubovoľne vytvorený blok je možné formulovať tvrdenie uvedené nižšie, ktorého využite sa ukáže v neskorších častiach tejto práce.

Tvrdenie 1.7. *Ak platí $D \subseteq Z$, potom ak je niektorý z útvarov množiny D neefektívny vzhľadom na množinu M^D generovanú blokom D, potom je neefektívny aj vzhľadom na množinu M generovanú množinou Z.*

Dôkaz. Nech platí $D \subseteq Z$, potom je zrejmé že vytvorený blok pozostáva z k útvarov

pričom platí, že $1 \leq k \leq n$. Ak by sme vzali $k = 1$, tak je zrejmé, že vybraný útvar je efektívny vzhľadom k množine produkčných možností generovanej ním samotným. Ak uvažujeme prípad, že $k = n$, potom takýto blok je tvorený všetkými pozorovanými útvarmi, teda platí $M^D = M$. Z toho je zrejmé, že tvrdenie platí triviálne. Uvažujme teda iba prípad, kedy $2 \leq k \leq n-1$. Predpokladajme, že existuje neefektívny útvar $(x_o, y_o) \in M^D$. Potom musí nutne existovať nejaký útvar $(x_d, y_d) \in M^D$, ktorý dominuje útvaru (x_o, y_o), teda platí, že $x_d \leq x_o$ a $y_d \geq y_o$ a zároveň $(x_o, y_o) \neq (x_d, y_d)$. Z konštrukcie množiny M^D je zrejmé, že platí $M^D \subseteq M$, keďže množina M je generovaná množinou Z, pozostávajúcou zo všetkých n útvarov. To však znamená, že útvar (x_d, y_d), ktorý dominuje útvaru (x_o, y_o), určite patrí množine M. Z toho však vyplýva, že útvar (x_o, y_o) je neefektívny taktiež vzhľadom k množine M, čo sme chceli dokázať. □

1.5 Aditívny model a superefektivita

Formulácie väčšiny DEA modelov sa viažu najmä k teórii lineárneho programovania, avšak existujú taktiež modernejšie modely, ktorých formulácie zodpovedajú úlohám nelineárneho programovania. Medzi zástupcov týchto modelov patria napr. Hyperbolický model ,prípadne Russelov model, ktoré sú formulované napr. v [16], [17]. V prípade modelov, formulovaných ako úloha LP, je zavedená základná schéma rozdelenia modelov na základe orientácie. Orientované modely sa zameriavajú na uprednostňovanie vstupov pred výstupmi ,resp. vice versa. Medzi základné orientované modely patria napríklad radiálne modely známe pod názvami CCR a BCC model, oba sformulované v dvoch variantoch a to vstupne a výstupne orientovaný model. Rozdiel modelov spočíva v uvažovaných výnosoch z rozsahu. CCR model uvažuje konštantné výnosy z rozsahu a naopak BCC model variabilné. V tejto práci sa nimi však nebudeme ďalej zaoberať, ale zdrojom základných informácii môžu byť napríklad publikácie [5], [2].

Súčasťou tejto diplomovej práce je neradiálny tzv. Aditívny model, sformulovaný v roku 1985 jeho autormi A. Charnesom, W.W. Cooperom, B. Golanym, L. Seifordom a J. Stutzom [4]. Tento model patrí do skupiny neorientovaných, čo znamená, že od predchádzajúcich spomenutých modelov sa líši v tom, že neuprednostňuje vstupy pred výstupmi prípadne výstupy pred vstupmi. Aditívny model tak nachádza uplatnenie práve v situáciách, keď nie je jednoznačne určené, či je v našom záujme maximalizácia produkcie

výstupov alebo minimalizácia spotreby vstupov.

1.5.1 Formulácia aditívneho modelu

Aditívny model môže byť sformulovaný pre konštantné výnosy z rozsahu ako aj pre variabilné výnosy z rozsahu. V našej práci sa obmedzíme práve na druhé spomenuté výnosy z rozsahu, teda variabilné. V aditívnom modeli vystupujú nezáporne doplnkové premenné s^x, s^y, nazývané slacky. Model je založený na maximálnej možnej projekcii útvarov na hranicu efektívnosti v smeroch klesajúcich vstupov a rastúcich výstupov, ktorým zodpovedajú spomenuté slacky. Môžeme tomu rozumieť aj nasledovným spôsobom. Aditívny model hľadá pre pozorovaný útvar DMU_o taký jeho efektívny vzor, ktorý maximalizuje ℓ_1 vzdialenosť od pozorovaného útvaru DMU_o. Aditívny obálkový model, vzhľadom k variabilným výnosom z rozsahu, môžeme v analytickom tvare zapísať ako [14]:

$$\max_{s^x,s^y} = \{e^T s^x + e^T s^y \mid (x_o - s^x, y_o + s^y) \in M_{VRS},\ \ s^x \geq 0_m, s^y \geq 0_s\}. \tag{6}$$

Následne uvedieme definíciu aditívneho modelu v obálkovom tvare odpovedajúcom variabilným výnosom z rozsahu pre konkrétne $DMU_o = (x_o, y_o)$, ako úlohu lineárneho programovania, pričom ako zdroj nám slúži [14].

Definícia 1.8 (Aditívny obálkový VRS model)**.** *Pod aditívnym modelom v obálkovom tvare odpovedajúcim variabilným výnosom z rozsahu, ozn. $AD-OM-VRS$ pre konkétne DMU_o pre $o \in \{1, \ldots, n\}$ rozumieme úlohu:*

$$\begin{aligned} \max_{\lambda,s^x,s^y}\ & A_O := e_m^T s^x + e_s^T s^y \\ s.t.\ & \sum_{j=1}^{n} x_j \lambda_j + s^x = x_o, \\ & \sum_{j=1}^{n} y_j \lambda_j - s^y = y_o, \\ & \sum_{j=1}^{n} \lambda_j = 1 \\ & \lambda \geq 0_n, s^x \geq 0_m, s^y \geq 0_s \end{aligned} \tag{7}$$

Optimálnu hodnotu účelovej funkcie budeme označovať A_O^*. Je zrejmé, že $A_O^* \in [0, \infty)$.

Dávame do pozornosti, že jediným rozdielom pre formuláciu úlohy aditívneho modelu v obálkovom tvare pre konštantné výnosy z rozsahu, by bolo vynechanie podmienky na

súčet lámbd, teda $\sum_{j=1}^{n} \lambda_j = 1$. Budeme hovoriť, že testovaný útvar DMU_o je efektívnym v zmysle aditívneho modelu práve vtedy, keď optimálna hodnota účelovej funkcie bude rovná nule $A_O^* = 0$, čo znamená, že pozorovaný útvar leží na hranici efektívnosti H_E. Inak budeme hovoriť, že testovaný útvar DMU_o je neefektívny. Podotýkame, že k efektívnosti testovaného útvaru DMU_o bude dochádzať iba v prípade, že optimálne hodnoty všetkých doplnkových premenných s^x a s^y budú rovné nule.

Ako sme už spomenuli, formulácie väčšiny DEA modelov patria k úlohám lineárneho programovania. Formulácia aditívneho modelu v obálkovej forme pre variabilné výnosy z rozsahu nie je výnimkou, a z tohto dôvodu k nej vieme zostrojiť príslušnú duálnu úlohu. Takúto formuláciu budeme nazývať aditívnym modelom v multiplikatívnom tvare odpovedajúcim variabilným výnosom z rozsahu. Pri nasledovnej formulácii modelu čerpáme z učebného textu [14].

Definícia 1.9 (Aditívny multiplikatívny VRS model)**.** *Pod aditívnym modelom v multiplikatívnom tvare odpovedajúcim variabilným výnosom z rozsahu, ozn. $AD-MM-VRS$ pre konkétne DMU_o pre $o \in 1, \ldots, n$ rozumieme úlohu:*

$$\begin{aligned} \min_{u,v,\sigma} \ & A_M := v^T x_o - u^T y_o + \sigma \\ s.t. \ & v^T x_j - u^T y_j + \sigma \geq 0, \ \ j = 1, \ldots, n, \\ & v \geq e_m, u \geq e_s, \ \ \sigma \in \mathbb{R} \end{aligned} \tag{8}$$

Optimálnu hodnotu účelovej funkcie budeme označovať A_M^*.

Zo silnej vety o dualite, ktorá je formulovaná napr. v [19] vieme, že ak existuje optimálne riešenie primárnej úlohy potom existuje aj optimálne riešenie jej duálneho náprotivku a hodnoty účelových funkcií sú rovnaké. Budeme teda hovoriť, že testovaný útvar DMU_o je efektívnym, v zmysle aditívneho modelu v multiplikatívnom tvare práve vtedy, keď optimálna hodnota účelovej funkcie $A_M^* = 0$. Inak budeme hovoriť, že testovaný útvar DMU_o je neefektívny. Poznamenajme, že jediným rozdielom pre formuláciu úlohy aditívneho modelu v multiplikatívnom tvare zodpovedajúceho konštantným výnosom z rozsahu by bolo vynechanie voľnej premennej σ.

1.5.2 Vlastnosti aditívneho modelu

Na úvod zavedieme pojem λ-vzoru pre aditívny model, ktorý sa bude vyskytovať v tvrdeniach obsiahnutých v nasledujúcej časti.

Definícia 1.10 ($\lambda - vzor$)**.** *Nech* $(x_o, y_o) \in M$ *a nech* $(\lambda^*, s^{x*}, s^{y*})$ *je optimálne riešenie úlohy* $AD - OM - VRS$*. Pod* λ*-vzorom bodu* (x_o, y_o) *v množine* M *rozumieme bod* $(x_o^{\lambda^*}, y_o^{\lambda^*}) \in M$ *taký, že platí:*

$$x_o^{\lambda^*} = \sum_{j=1}^{n} x_j \lambda_j^* = x_o - s^{x*}, y_o^{\lambda^*} = \sum_{j=1}^{n} y_j \lambda_j^* = y_o + s^{y*} \tag{9}$$

Vlastnosti aditívneho modelu so sebou prinášajú mnoho výhod. Jedna v tejto práci využívaná výhoda použitia aditívneho modelu spočíva v tom, že nemusíme uvažovať oproti radiálnym modelom, či optimálna hodnota účelovej funkcie predstavuje pseudoefektivitu alebo efektivitu, keďže vzorom útvaru je vždy útvar z hranice efektívnosti. Táto vlastnosť nie je výsadou iba aditívneho modelu ale napríklad aj tzv. SBM modelu, ktorý je formulovaný napr. v [23]. Teraz uvedieme formuláciu vety s dôkazom, ktorá hovorí o uvedenej vlastnosti aditívneho modelu.

Tvrdenie 1.11. λ*-vzor pri aditívnom modeli je vždy útvar na efektívnej hranici.*

Dôkaz. Dôkaz tvrdenia prevedieme sporom. To znamená, že predpokladajme, že existuje λ-vzor pri aditívnom modeli, ktorý neleží na efektívnej hranici. Platí, že λ-vzor útvaru DMU_o pri aditívnom modeli je optimálnym riešením úlohy lineárneho programovania $AD-OM-VRS$. Nech pre útvar DMU_o je $(\lambda^*, s^{x*}, s^{y*})$ optimálne riešenie danej úlohy s optimálnou hodnotou účelovej funkcie $A_O^* = e_m^T s^{x*} + e_s^T s^{y*}$. Nech teda λ-vzor $(x_o^{\lambda^*}, y_o^{\lambda^*})$ je neefektívny útvar. Potom je možné znížiť niektorú hodnotu jeho vstupu $x_o^{\lambda^{**}} = x_o^{\lambda^*} - s^{x**}$, pre $s^{x**} > 0$, resp. zvýšiť niektorú hodnotu jeho výstupu $y_o^{\lambda^{**}} = y_o^{\lambda^*} + s^{y**}$, pre $s^{y**} > 0$, pričom takýto útvar $(x_o^{\lambda^{**}}, y_o^{\lambda^{**}})$ stále leží v množine produkčných možností M. Každá operácia zníženia vstupu ,resp. zvýšenia výstupu však nutne vedie k zvýšeniu hodnoty účelovej funkcie. To je však v spore s predpokladom jej optimálnosti, čo znamená, že platí pôvodné tvrdenie, teda λ-vzor pri aditívnom modeli je vždy útvar na efektívnej hranici. □

Okrem Tvrdenia 1.11, ktoré hovorí, že λ-vzor aditívneho modelu je útvar z hranice efektívnosti, vieme pre λ-vzor formulovať ďalšie tvrdenie, ktoré uvádzame nižšie.

Tvrdenie 1.12. λ*-vzor pri aditívnom modeli, vzhľadom k variabilným výnosom z rozsahu, je vždy konvexnou kombináciou efektívnych útvarov.*

Dôkaz. Pre každé $j \in \{1, \ldots, n\}$ platí, že komplementárnou premennou k ohraničenej premennej λ_j je doplnková premenná pre nerovnosť $u^T y_j - v^T x_j + \sigma \leq 0$. Označme túto doplnkovú premennú d_j, pričom $d_j \geq 0$. To znamená, že platí $u^T y_j - v^T x_j + \sigma + d_j = 0$. V optimálnom riešení λ_j^* a (u^*, v^*, z^*, d_j^*) platí $\lambda_j^* d_j^* = 0$. Z komplementarity dostávame, že ak $\lambda_j^* > 0$ potom nutne musí byť $d_j^* = 0$ a teda platí $u^T y_j - v^T x_j + \sigma = 0$. Nadrovina $\{(x, y) | u^T y - v^T x + \sigma = 0\}$ je opornou nadrovinou množiny M_{VRS}, pričom platí $(x_j, y_j) \in M_{VRS}$, čo znamená, že bod (x_j, y_j) patrí hranici množiny M_{VRS}. Tento postup môžeme použiť pre ľubovoľný bod, ktorý prislúcha kladnej premennej λ_j. To znamená, že všetky body, ktorým prislúchajú kladné premenné λ, budú súčasťou danej opornej nadroviny a teda budú ležať na hranici množiny M_{VRS}. Z optimálneho riešenia je zrejmý aj normálový vektor opornej nadroviny, ktorý je rovný (u^*, v^*). Z ohraničení v multiplikatívnom tvare aditívneho modelu pre variabilné výnosy z rozsahu je zrejmé, že normálový vektor opornej nadroviny má každú zložku väčšiu alebo rovnú 1. To však znamená, že oporná nadrovina, daná rovnicou $u^T y - v^T x + \sigma = 0$ je rôznobežná s opornými nadrovinami tvoriacimi hranicu pseudoefektívnosti množiny M_{VRS}, a teda sa jedná o opornú nadrovinu prechádzajúcu hranicou efektívnosti. To však znamená, že útvary ležiace v danej nadrovine a patriace hranici efektívnosti množiny M_{VRS} sú nutne efektívne. Z podmienky konvexnosti $\sum_{j=1}^{n} \lambda_j = 1$ dostávame požadované tvrdenie vety. □

Dôsledkom vyššie uvedeného tvrdenia je, že pri vyhodnocovaní efektivity neefektívnych útvarov stačí vyhodnocovať efektivitu vzhľadom na množinu všetkých efektívnych útvarov. Tento dôsledok vedie ku konštrukcii algoritmu, ktorý bude predstavený v nasledujúcej kapitole.

Aditívny model je považovaný pravdepodobne za najjednoduchšieho zástupcu neorientovaných modelov. Optimálnu hodnotu účelovej funkcie aditívneho modelu, ako sme ju uviedli v definíciách úloh $AD-OM-VRS$,resp. $AD-MM-VRS$, nemôžeme rozumieť ako technickú efektivitu. Je tomu tak z dôvodu, že nenapĺňa predpoklad jej vyjadriteľnosti v intervale $(0, 1]$. Na základe optimálnej hodnoty účelovej funkcie však vieme jednoznačne rozhodnúť o klasifikácii útvaru na efektívny ,resp. neefektívny. Z formulácie účelovej funkcie a z dôvodu, že množina prípustných riešení vytvára neohraničený polyéder, môžeme ľahko dôjsť k zisteniu, že hodnoty optimálnych riešení účelovej funkcie sa nachádzajú v intervale $[0, \infty)$. Z tohto dôvodu je potreba istej dodatočnej modifikácie účelovej fun-

kcie, ktorá by už spĺňala požadovaný predpoklad. Zdrojom informácii rôznych modifikácii účelovej funkcie aditívneho modelu môže byť napr. [14] a [15].

Účelová funkcia aditívneho modelu má vlastnosť monotónnosti, ktorá je presne sformulovaná a dokázaná v [15]. Tejto vlastnosti môžeme rozumieť takým spôsobom, že ak dôjde pre konkrétny pozorovaný neefektívny útvar k zvýšeniu niektorých jeho vstupov bez zmeny hodnôt výstupov, prípadne opačne, teda že dôjde k zníženiu niektorých hodnôt výstupov bez zmeny vstupných hodnôt, táto zmena nutne povedie k poklesu optimálnej hodnoty účelovej funkcie.

Základný aditívny model nie je invariantný na zmenu jednotiek, avšak transformácia ponecháva klasifikáciu útvarov na efektívne resp. neefektívne útvary. Existujú isté modifikácie, ktoré vedú k tzv. aditívnemu modelu s váhami, ktoré sú spracované napr. v [15], a tieto už sú invariantné na zmenu jednotiek. Model je invariantný na posun, čo nachádza uplatnenie najmä v prípadoch, keď nie je splnený základný predpoklad nezápornosti dát.

Pravdepodobne najväčšou výhodou aditívneho modelu je, že vyriešením jedinej úlohy lineárneho programovania, bez potreby riešenia doplnkovej úlohy ako u orientovaných modeloch, sme schopný klasifikácie pozorovaného útvaru na efektívny ,resp. neefektívny. Táto vlastnosť vychádza z Tvrdenia 1.11, ktoré hovorí, že λ-vzor pozorovaného útvaru je vždy z hranie efektívnosti H_E. Nespornou výhodou aditívneho modelu je jeho jednoduchá transformácia na model, ktorému sa venujeme v nasledovnej časti a ktorý tvorí základ konštrukcie jedného z algoritmov vedúcich k redukcii výpočtového času.

1.5.3 Superefektivita pre aditívny model

V tejto časti práce sa budeme zaoberať novým pojmom obálkovej analýzy dát, ktorým je tzv. superefektivita, ktorá býva zvyčajne ozn. SE. Superefektivita predstavuje nástroj, ktorý pomáha rozlišovať medzi efektívnymi producentmi. V jednoduchosti môžeme superefektivite rozumieť takým spôsobom, že sa jedná o istú mieru vyjadrujúcu zmenu aproximačnej množiny produkčných možností M vynechaním jedného z pozorovaných útvarov.

Konštrukcie superefektivity pri orientovaných modeloch sú priamočiare, keďže jediným rozdielom oproti pôvodným modelom je vynechanie pozorovaného útvaru z množiny Z, útvarov generujúcich množinu produkčných možností M, čo sa prejaví na λ-súčtoch. Zvyšok

je totožný s formuláciami pôvodných modelov. Je potrebné si uvedomiť, že v prípade konštantných výnosov z rozsahu je superefektivita orientovaných modelov dobre definovaná, avšak v prípade variabilných výnosov z rozsahu môže dochádzať k istým problémom.

Formulácia superefektivity, v prípade nami zvoleného aditívneho modelu, je mierne zložitejšia, avšak stále dostatočne priamočiara. Bola formulovaná a zdôvodnená pre všeobecný prípad neradiálnych neorientovaných modelov v [15]. Konkrétne, v prípade aditívneho modelu, sa s formuláciou superefektivity môžeme stretnúť napr. v publikácii [6]. Hlavými rozdielmi sú, oproti formuláciám superefektivity v prípadoch orientovaných modelov, že nestačí jednoducho vynechať pozorovaný útvar z množiny produkčných možností ako v prípade orientovaných modelov, ale je taktiež potreba zmeny maximalizácie účelovej funkcie, pozostávajúcej zo súčtu slackov, na minimalizáciu. Ďalšími potrebnými transformáciami sú zmena znamienok pred slackmi vystupujúcimi v ohraničeniach a taktiež zmeny rovností v podmienkach prípustnosti na nerovnosti, ktoré korešpondujú s nerovnosťami ako v prípade orientovaných modelov. Po vykonaní všetkých potrebných transformačných úprav dochádzame k definícii aditívneho modelu pre superefektivitu, ktorú uvádzame nižšie.

Definícia 1.13 (Aditívny obálkový VRS model pre superefektivitu)**.** *Pod aditívnym modelom v obálkovom tvare odpovedajúcim variabilným výnosom z rozsahu pre výpočet superefektivity, ozn.* $AD-OM-VRS-SE$ *pre konkétne* DMU_o *pre* $o \in 1, \ldots, n$ *rozumieme úlohu:*

$$\begin{aligned} \min_{\lambda, s^x, s^y} \quad & A_O^{SE} := e_m^T s^x + e_s^T s^y \\ s.t. \quad & \sum_{j=1, j\neq o}^{n} x_j \lambda_j - s^x \leq x_o \\ & \sum_{j=1, j\neq o}^{n} y_j \lambda_j + s^y \geq y_o \\ & \sum_{j=1 j\neq o}^{n} \lambda_j = 1 \\ & \lambda \geq 0_n, s^x \geq 0_m, s^y \geq 0_s \end{aligned} \tag{10}$$

Optimálnu hodnotu účelovej funkcie budeme označovať A_O^{SE*}.

Z formulácie úlohy je zrejmé, že optimálna hodnota účelovej funkcie pre pozorovaný útvar DMU_o je $A_O^{SE*} \in [0, \infty)$. Optimálna hodnota účelovej funkcie uvedenej úlohy predstavuje ℓ_1 vzdialenosť pozorovaného útvaru DMU_o od redukovanej množiny M, vytvorenej

zo zvyšných útvarov. Zároveň platí, že ak jej hodnota bude kladná, potom nutne platí, že pozorovaný útvar je efektívny. Dávame však do pozornosti, že opačná implikácia neplatí, čiže ak je útvar efektívny, nemusí nutne vyjsť optimálna hodnota účelovej funkcie kladná. K tomuto prípadu dochádza v prípadoch, keď efektívny útvar leží vo vnútri steny prípadne hrany konvexného polyédra tvoriacej hranicu efektívnosti H_E. Z dôvodu vylúčenia útvaru, pre ktorý optimálna hodnota účelovej funkcie je kladná, sa takýto útvar ocitne mimo redukovanej množiny produkčných možností M, čo je ekvivalentné s tvrdením, že vylúčením takéhoto útvaru sa mení hranica pôvodnej množiny produkčných možností ∂M. Dodajme, že množina všetkých útvarov, ktorých vynechanie spôsobí zmenu hranice ∂M je najmenšia možná množina útvarov, ktorá jednoznačne určuje pôvodnú množinu produkčných možností M. Na poslednom tvrdení je postavený základ algoritmu vedúci k zrýchleniu výpočtu, ktorému sa budeme venovať v neskorších častiach tejto práce. Duálnym modelom k modelu $AD-OM-VRS-SE$ je model, ktorého definíciu uvádzame nižšie.

Definícia 1.14 (Aditívny multiplikatívny VRS model pre superefektivitu)**.** *Pod aditívnym modelom v multiplikatívnom tvare odpovedajúcim variabilným výnosom z rozsahu pre výpočet superefektivity, ozn.* $AD-MM-VRS-SE$ *pre konkétne* DMU_o *pre* $o \in 1,\ldots,n$ *rozumieme úlohu:*

$$\begin{aligned} \max_{u,v,\sigma}\ & A_M^{SE} := u^T y_o - v^T x_o + \sigma \\ s.t.\ & u^T y_j - v^T x_j + \sigma \leq 0, j \in \{1,\ldots,n\} \setminus o \\ & u \leq e_s \\ & v \leq e_m \\ & u \geq 0, v \geq 0 \end{aligned} \tag{11}$$

Optimálnu hodnotu účelovej funkcie budeme označovať A_M^{SE*}.

2 Algoritmy znižujúce výpočtovú náročnosť

V nadväznosti na skutočnosť, že v posledných rokoch sme svedkami obrovského technologického pokroku, ktorý so sebou na jednej strane prináša neustále zlepšenia výpočtovej sily našich zariadení, avšak na strane druhej nedozerne narastajú aj množstvá gigantických datasetov, ktoré môžu byť podrobené rozličným analýzam, je potreba implementácie nových postupov zrýchľujúcich výpočtový čas riešenia úloh.

2.1 Motivácia

Hoci sa v obálkovej analýze dát zatiaľ významne neobjavujú veľké dátové súbory, môžeme naprieč literatúrou venujúcou sa problematike DEA už v súčasnosti pozorovať prípady štúdií, kde boli analyzované veľké dátové sety. V roku 1997 autori Richard S. Barr a Matthew L. Durchloz publikovali článok [3], ktorého súčasťou bol dátový set pozostávajúci z 8748 pozorovaní bánk, pričom každý pozorovaný útvar pozostával zo 6 vstupov a 3 výstupov. Jednalo sa v tej dobe o jeden z najväčších dátových setov, pričom evalvácia s vtedajšou výpočtovou silou techniky si vyžiadala približne 44.21 hodín. Ďalším príkladom veľkej dátovej sady bola štúdia autorov Thomas F. Siemsa a Richard S. Barra [20] z roku 1998 obsahujúca 11379 pozorovaní bánk s 5 vstupmi a 3 výstupmi. Autori Dariush Khezrimotlagh, Joe Zhu, Wade D. Cook a Mehdi Toloo v publikácii [18] z roku 2018 spomínajú dátovú sadu, pozostávajúcu z 30099 pozorovaní elektrární s 2 vstupmi a 4 výstupmi. Na internetovej stránke [9] môžeme nájsť dáta pozostávajúce z 19939 pozorovaní komerčných bánk so 6 vstupmi a 5 výstupmi. Hoci prístupných dátových sád veľkých rozmerov, pričom podľa [18] sa za veľké dáta považujú sety obsahujúce 20000 DMU a viac, vhodných na obálkovú analýzu dát v súčasnosti nemusí byť dostatok, odborníci, možno aj pod vplyvom vývoja v iných disciplínach, sa snažia vyvíjať efektívnejšie algoritmy vedúce k redukcii celkového výpočtového času.

Aplikovať určitý DEA model na danú sadu dát n útvarov s m vstupmi a s výstupmi, znamená postupne riešiť sadu n úloh LP totožného rozmeru, teda rozmeru $n \times (m + s)$. Na riešenia týchto úloh LP možno použiť rôzne metódy, napríklad simplexovú metódu alebo metódu vnútorného bodu. Druhá zmienená metóda bude využitá aj v tejto práci. Príslušné úlohy LP v danej sade o n útvaroch majú rovnakú účelovú funkciu a zároveň

rovnaké ľavé strany v podmienkach prípustnosti, pričom jediný rozdiel tkvie v odlišných formuláciách ohraničení na pravej strane. Znamená to teda, že sa pre každý pozorovaný útvar DMU_o, pre $o \in \{1, \ldots, n\}$, zmení pravá strana týchto ohraničení zodpovedajúca vstupom a výstupom pozorovaného útvaru. V zmysle výpočtovej zložitosti problému to znamená, že každá úloha má rovnaký rozmer a teda každá úloha formulovaná pre jednotlivé pozorované útvary DMU_o pre $o \in \{1, \ldots, n\}$ je, čo sa týka časovej náročnosti výpočtu, ekvivalentná. Takýto prístup k riešeniu problému sa naprieč literatúrou, týkajúcou sa obálkovej analýze dát, označuje za tradičný. Tradičný prístup je možný a zároveň postačujúci za predpokladu, že zadané dátové sady nie sú príliš veľké. To znamená, že pokiaľ celkový počet útvarov $|Z|$ v dátovej sade, nazývaný v súlade s terminológiou obálkovej analýze dát ako *kardinalita* úlohy, prípadne rozmery vektorov vstupov a výstupov m,s nazývané ako *dimenzia* úlohy, sú relatívne nízke.

Akonáhle signifikantne stúpne ktorákoľvek z prvých dvoch charakteristík dátovej sady, teda narastie celkový počet útvarov n, prípadne sa zväčší rozmer vektora vstupov m a výstupov s, výpočtový čas využívajúci tradičný prístup riešenia problému prudko narastie. Tento nárast následne povedie k tomu, že riešenia DEA modelov začínajú mať problémy s nájdením optimálneho riešenia úlohy v rozumnom čase. V takýchto prípadoch sa vynára potreba použitia určitého zlepšenia, ktoré má za cieľ podstatné zníženie výpočtovej náročnosti daných problémov. Ak by sme disponovali veľkými výpočtovými zdrojmi, čomu rozumieme tak, že by sme mali k dispozícii dostatok počítačov, na ktorých by mohli byť vykonávané výpočty simultánne, mohlo by sa naskytnúť postačujúce riešenie daného problému. Riešenie by pozostávalo v rozdelení útvarov, na približne veľkostne ekvivalentné množiny, a výpočet by bol následne vykonaný na všetkých počítačoch, ktoré máme k dispozícii, simultánne. Vyššie popísaný proces výpočtu sa nazýva *paralelizácia*. Tento prístup však častokrát nemusí byť uskutočniteľný, keďže nebýva bežné, že máme splnený predpoklad dispozície väčších výpočtových zdrojov. Ak by sme však aj mali k dispozícii viacero počítačov, ktoré by sme mohli simultánne využívať, nemusela by byť ani samotná paralelizácia výpočtu postačujúca. Ak by sme predpokladali, že naša výpočtová kapacita narastie 10-násobne, avšak dátová sada, ktorú máme podrobiť obálkovej analýze dát narastie 100-násobne tak sa opätovne dostávame na počiatok problému. Dôsledkom možného zlyhania tohto prístupu je potreba vývoja efektívnejších postupov výpočtu. Pro-

ces paralelizácie sa však ukazuje ako jedno z vhodných riešení problému týkajúceho sa DEA modelov v prípadoch veľkých dátových setov. Nie však takým spôsobom, ako vo vyššie uvedenom triviálnom príklade, ale ako súčasť ďalších úvah, ktoré boli predstavené v [3], a ktorým sa budeme viac venovať v podkapitole 2.2.2.

Poslednou, avšak nemenej dôležitou charakteristikou týkajúcou sa dátovej sady v zmysle DEA modelov je pomer počtu efektívnych útvarov k celkovému počtu útvarov. Táto charakteristika sa zvyčajne označuje ako *hustota*, pričom sa jedná o relatívnu mieru, ktorá je spravidla udávaná v percentách. Práve hustota dátovej sady, ako aj uvidíme v neskorších častiach tejto práce, sa ukazuje ako kľúčová charakteristika vedúca k pomerne vysokému zrýchleniu výpočtu. Najmä v prípadoch veľkých dátových sád, nízka hustota vedie k nemalým časovým úsporám pri riešeniach DEA modelov. Dodávame, že hustota dátovej sady nie je častokrát známa, avšak bolo by ju možné odhadnúť náhodným výberom niekoľkých pozorovaných útvarov a následným vyriešením príslušných úloh LP pre zvolené útvary. V článku [7] autora J.H. Dulá sa dátové sady na základe hustoty delia na 3 skupiny. Dáta s 1% efektívnych útvarov sa označujú ako dáta s nízkou hustotou, okolo 10% ako dáta so strednou hustotou a okolo 25% ako dáta s vysokou hustotou. V článku [18] sa však môžeme stretnúť s tvrdením, že za dáta s veľkou hustotou sa môžu považovať dátové sady, ktoré majú už 10% efektívnych útvarov. Pre reálne dátové sady však platí, že vyššie hodnoty hustôt sú zriedkavé, pričom obzvlášť pre veľké dátove sady sú časté najmä nižšie hodnoty hustôt [13].

2.2 Prehľad algoritmov

Úlohy, pri ktorých počty pozorovaných DMU nepresahujú úroveň niekoľko tisíc, je možné riešiť tradičným iteratívnym spôsobom. Dodávame, že obálková analýza dát je v prípade viac ako niekoľko tisíc DMU považovaná za časovo náročnú metódu. Z týchto dôvodov sa v ostatných rokoch čoraz viac začínajú vynárať štúdie navrhujúce isté zlepšenia vedúce k časovým úsporám. Všetky, v tejto kapitole predstavené zlepšujúce algoritmy, sú založené na akomsi princípe efektívnejšieho využitia informácie o optimálnych riešeniach predchádzajúcich úloh. Následné zužitkovanie získanej informácie vedie, pri formuláciách modelov pre zatiaľ neohodnotené útvary, k časovým úsporám. To znamená, že uvedené algoritmy budú taktiež fungovať iteratívnym spôsobom riešenia sady úloh LP, avšak sa budú snažiť

znížiť kardinalitu úlohy na základe informácií z predošlých optimálnych riešení úloh. V nasledovnej časti tejto diplomovej práce si preto predstavíme pokročilejšie algoritmy, ktoré istým spôsobom využívajú informácie o optimálnych riešeniach predchádzajúcich úloh LP, a vedú k značnému zníženiu výpočtovej zložitosti DEA modelov, pričom pri spracovaní tejto kapitoly sme vychádzali najmä z publikácií [1], [3], [8], [18] ale taktiež z článkov [21], [22], [7], [10].

2.2.1 Restricted basis entry metóda

Ako sme v úvode tejto kapitoly naznačili, tradičný výpočtový prístup v obálkovej analýze dát spočíva v riešení úloh lineárneho programovania pre každé DMU_j pre $j \in \{1, \ldots, n\}$ samostatne, pričom informácie o klasifikácii útvarov na efektívne, resp. neefektívne útvary získané z optimálnych riešení úloh LP, prípadne informácie o efektívnych vzoroch pre jednotlivé DMU_j, sa nijakým spôsobom nezohľadňujú pri konštrukciách žiadnej ďalšej úlohy. To znamená, že všetky nasledujúce úlohy majú rovnakú kardinalitu n a dimenziu $m + s$ ako predchádzajúce, resp. ako základná úloha LP, a teda rozmer úlohy $n \times (m + s)$ ostáva nemenný.

Prvá, v tejto podkapitole predstavená metóda, znižujúca celkový výpočtový čas, nazývaná RBE (z angl. *Restricted basis entry*), bola prezentovaná roku 1993 Agha I. Alim [1]. Táto metóda je založená na akomsi princípe, ktorý zohľadňuje výsledky optimálneho riešenia predchádzajúcej úlohy LP takým spôsobom, že pre všetky nasledujúce úlohy zníži kardinalitu oproti predchádzajúcim úlohám za predpokladu, že pozorovaný útvar je vyhodnotený ako neefektívny.

Predtým, ako prejdeme k samotnému popisu algoritmu, je potrebné ešte zaviesť pojem *bázy*. Pod bázou v tejto metóde budeme rozumieť aktuálnu podmnožinu množiny Z, vzhľadom na ktorú počítame mieru efektívnosti daného DMU_j. Poznamenávame, že na počiatku začíname s bázou, ktorá je rovná celému Z. To znamená, že sa v nej nachádzajú všetky pozorované útvary. Podstata metódy RBE spočíva v postupnej redukcii tejto bázy o útvar, ktorý bol vyhodnotený ako neefektívny vzhľadom na aktuálnu bázu.

Keďže platí, že neefektívny útvar sa nachádza nutne mimo hranice efektívnosti množiny produkčných možností, jeho príslušná λ_j sa nikdy nebude spolupodieľať na optimálnych riešeniach žiadnych ďalších útvarov 1.12. To znamená, že v optimálnych riešeniach

úloh LP pri klasifikáciách zvyšných útvarov bude vždy rovná 0, keďže sa nebude vyskytovať v efektívnom vzore pre žiaden útvar. Následkom toho je, že koeficient λ_j prislúchajúci danému neefektívnemu útvaru môže byť pre nasledovné iterácie z bázy odstránený.

Tento jednoduchý postup bude viesť pri nasledujúcej iterácii k redukcii predošlej úlohy na novú úlohu LP, ktorá už pri optimalizácii nebude narábať s bázou pozostávajúcou zo všetkých pôvodných útvarov, ale iba s jej podmnožinou vytvorenou vynechaním daného neefektívneho útvaru. Je dobré si uvedomiť, že týmto jednoduchým princípom tak znížime kardinalitu všetkých nasledujúcich nevyriešených úloh LP o jedna. Tento proces sa opakuje pre ďalšie neklasifikované útvary, až pokým nie sú z bázy odstránené všetky neefektívne útvary, a teda v báze ostanú iba útvary efektívne. Ako sa ukazuje, tento jednoduchý princíp vedie k signifikantnému zníženiu výpočtového času, ktorý bude demonštrovaný v poslednej kapitole.

Pre lepšie pochopenie spomínaného algoritmu, prezentujeme nižšie 4-krokový pseudokód. Ešte predtým je však potrebné uviesť symboliku použitú v nasledovnom pseudokóde. V prvom kroku, ktorý označujeme ako *inicializácia*, budeme symbolom D_Z označovať množinu indexov všetkých DMU_j pre $j \in \{1, \ldots, n\}$. Indexová množina D_Z obsahuje indexy útvarov vystupujúcich v podmienkach prípustnosti v jednotlivých úlohách LP, teda jedná sa o indexy útvarov prislúchajúcich báze. To znamená, že v prvej iterácii sa v podmienkach prípustnosti objaví všetkých n útvarov, keďže žiaden nebol doposiaľ klasifikovaný. Symbolom D_E budeme následne označovať množinu indexov efektívnych útvarov, pričom vo fáze inicializácie je D_E prázdnou množinou, keďže žiaden z pozorovaných útvarov nebol doposiaľ klasifikovaný ako efektívny. Napokon symbolom D_I budeme označovať množinu indexov neefektívnych útvarov, pričom tak ako v prípade množiny D_E, je množina D_I v kroku inicializácie prázdnou množinou.

V druhom kroku riešime príslušnú úlohu LP, pričom v podmienkach prípustnosti prechádzame množinou D_Z, ktorá prislúcha indexom obsiahnutým v aktuálnej báze.

V treťom kroku, ktorý označujeme ako *aktualizácia*, zatrieďujeme útvary na základe optimálnych hodnôt úlohy LP z predchádzajúceho kroku. V prípade, že optimálna hodnota účelovej funkcie v druhom kroku bola rovná 0, čo znamená, že v zmysle aditívneho modelu sa jedná o efektívny útvar, index zodpovedajúci pozorovanému útvaru je zaradený do množiny indexov efektívnych útvarov D_E, pričom množina D_Z ostáva v nasledovnej

iterácii nezmenená, keďže, ako sme uviedli, takýto útvar sa môže spolupodieľať na vytvorení efektívneho vzoru niektorých nasledovných útvarov. Ak však optimálna hodnota účelovej funkcie nie je rovná 0, čo pri použití aditívneho modelu znamená, že sa jedná o neefektívny útvar, index útvaru je zaradený do množiny indexov neefektívnych útvarov D_I a zároveň je príslušný index odstránený z množiny D_Z. V tomto kroku teda dochádza k redukcii množiny D_Z pre nasledovné iterácie. Z množiny D_Z sú teda postupnými iteráciami odstraňované indexy zodpovedajúce neefektívnym útvarom. To znamená, že sa v nej v jednotlivých iteráciách nachádzajú buď indexy útvarov, ktoré boli vyhodnotené ako efektívne v predošlých iteráciách alebo indexy doposiaľ ešte neohodnotených útvarov. Tak ako sme na začiatku uviedli, rozmer úlohy postupne klesá a platí, že v prvej iterácii je rozmer úlohy rovný $n \times (m + s)$ a ak bol útvar klasifikovaný ako neefektívny, tak sa zníži kardinalita o 1, teda rozmer pre nasledovnú iteráciu bude rovný $(n \dot{-} 1) \times (m + s)$.

V poslednom kroku je vykonaná kontrola počtu iterácií. Ak je počet iterácií nižší ako n, teda doposiaľ neboli ohodnotené všetky útvary, prejdeme opätovne ku kroku 2 a celý proces opakujeme, pokým nevyhodnotíme všetky útvary z dátovej sady.

RBE pseudokód:

Krok 1 Inicializácia.

$Z = \{DMU_1, \ldots, DMU_n\}$

$D_E \leftarrow \emptyset$

$D_I \leftarrow \emptyset$

$D_Z = \{1, \ldots, n\}$

$o = 1$

Krok 2 Riešenie úlohy LP, pre dané $DMU_o \in Z$

$$\begin{aligned} \max_{\lambda, s^x, s^y} \; & A_o := e_m^T s^x + e_s^T s^y \\ s.t. \; & \sum_{j \in D_Z} x_j \lambda_j + s^x = x_o, \\ & \sum_{j \in D_Z} y_j \lambda_j - s^y = y_o, \\ & \sum_{j \in D_Z} \lambda_j = 1 \\ & \lambda \geq 0_{|D_Z|}, s^x \geq 0_m, s^y \geq 0_s \end{aligned} \tag{12}$$

Krok 3 Aktualizácia.

Ak útvar DMU_o je klasifikovaný ako efektívny, potom:

$D_E \leftarrow D_E \cup \{o\} \quad \wedge \quad D_Z = D_Z,$

inak:

$D_I \leftarrow D_I \cup \{o\} \quad \wedge \quad D_Z = D_Z \setminus \{o\}$

Krok 4 Inkrementácia.

Ak $o < n$, potom $o = o + 1$ a naspäť ku Kroku 2 pre ďalšie DMU_o, inak KONIEC.

Pre vyššie uvedený algoritmus existujú ďalšie možnosti jeho zlepšenia. V článku [1] je uvedená Lemma 1, ktorá hovorí, že pokiaľ existuje v množine pozorovaných útvarov dominovaný útvar, ktorý je dominovaný niektorým iným útvarom, potom takýto útvar je nutne neefektívny. Je dobré si uvedomiť, že množina všetkých dominovaných útvarov môže byť z modelu dočasne odstránená, keďže sa jedná o neefektívne útvary. Týmto postupom môžeme znížiť kardinalitu úlohy o počet dominovaných útvarov, bez potreby riešenia úlohy LP. Je zrejmé, že čím je počet dominovaných útvarov vyšší, tým väčšie úspory času môžeme očakávať. Je potrebné podotknúť, že efektivity dominovaných útvarov nám nie sú známe pred ich dočasným odstránením z dátovej sady. Z tohto dôvodu je potrebné zistiť efektivitu takýchto útvarov, ktorá bude vykonaná na konci algoritmu RBE, keď už bude množina efektívnych útvarov známa. Následne sa postupne vyriešia úlohy LP, prislúchajúce dominovaným útvarom, kde sa v podmienkach prípustnosti budú vyskytovať iba efektívne útvary.

V roku 1989 autori T. Sueyoshi a Y. Chang publikovali článok [22], v ktorom navrhovali postup zrýchlenia výpočtu, ktorý v prvej fáze využíval rozdelenie dátovej sady na dominované a nedominované útvary. Postup autorov využíval k výpočtu revidovanú simplexovú metódu, ktorú v tejto práci neuvádzame, ale ako zdroj informácií môže slúžiť napr. [19].

Všetky predchádzajúce úvahy využívali pôvodné usporiadanie dát. T. Sueyoshi roku 1990 publikoval článok [21], ktorý na určenie dominovaných útvarov využíva preusporiadané dáta. Pôvodné dáta sú preusporiadané na základe tzv. *crude efficiency indicator*, ktorý je definovaný ako:

$$g_j = \frac{e_s^T y_j}{e_m^T x_j} = \frac{\sum_{r=1}^{s} y_{rj}}{\sum_{i=1}^{m} x_{ij}} \qquad j \in \{1, \ldots, n\}. \tag{13}$$

V spomenutom článku sa k určeniu množiny dominovaných používa zostupné zoradenie útvarov na základe indikátora g_j, pričom autor článku to zdôvodňuje tým, že útvary s vyššou hodnotou indikátora majú väčšiu šancu, aby dominovali niektorému z útvarov, čo vedie k časovým úsporám. Navyše podľa Lemmy 3 z článku [1], útvar s najvyššou hodnotou indikátora je efektívny. Nedominované útvary sú naopak zoradené na základe indikátora zostupne. Útvary s nižšími hodnotami indikátora, majú podľa [21] väčšiu pravdepodobnosť, že budú neefektívne a tým pádom budú môcť byť z modelu odstránené skôr.

V tejto diplomovej práci využívame ku klasifikácii útvarov aditívny model. Tento model disponuje vlastnosťou, ktorá dáva pre účely tejto diplomovej práce výhodu oproti použitiu orientovaných modelov CCR príp. BCC. Na základe Tvrdenia 1.11 uvedeného v kapitole 1, aditívny model projektuje pozorovaný útvar na hranicu efektívnosti, čo je rozdiel oproti orientovaným modelom, pri ktorých môže dôjsť k projekciám na hranicu pseudoefektívnosti. To znamená, že pre využitie algoritmu RBE, je potreba riešiť doplnkovú úlohu v prípade použitia orientovaných modelov. Tvrdenie 1.12, uvedené v kapitole 1 hovorí, že λ-vzor pozorovaného útvaru pri aditívnom modeli, vzhľadom k variabilným výnosom z rozsahu, je konvexnou kombináciou efektívnych útvarov.

Práve toto tvrdenie dáva možnosť ďalšieho vylepšenia uvedeného algoritmu. Zlepšenie algoritmu je možné takým spôsobom, že ak bol útvar vyhodnotený ako neefektívny v zmysle aditívneho modelu, potom z optimálneho riešenia príslušnej úlohy môžeme vziať všetky také útvary DMU_j, pre ktoré ich príslušné premenné λ_j vyšli kladné, keďže takéto útvary sú jednoznačne efektívne. Pre každý takto zistený útvar už nemusíme riešiť príslušnú úlohu LP, keďže z definície optimálnej hodnoty účelovej funkcie je zrejmé, že bude rovná 0. Tento postup môže viesť k ďalšiemu zníženiu celkového výpočtového času.

Ak navyše využijeme vzostupné preusporiadanie na základe indikátora spomenutého vyššie, je pravdepodobné, že neefektívne útvary budú vyhodnotené skôr, čo zvýši pravdepodobnosť určenia efektívnych útvarov bez potreby riešení úloh LP. Navyše kardinalita úlohy je znižovaná skôr, z dôvodu skorších klasifikácií neefektívnych útvarov, čo môže taktiež prispieť k časovým úsporám.

2.2.2 Metóda hierarchickej dekompozície

Ďalším predstaveným algoritmom, zrýchľujúci výpočet v obálkovej analýze dát a teda vedúci k značnému zníženiu celkového výpočtového času, je algoritmus takzvanej *hierarchickej dekompozície* (z angl. *Hierarchical Decomposition*). Tento algoritmus bol predstavený roku 1997 jeho autormi R.S. Barrom a M.L Durchholzom [3]. Hlavnou myšlienkou navrhovaného algoritmu ako už zo samotného názvu vyplýva, bude postupné rozdelenie množiny Z obsahujúcej všetky pozorované útvary DMU_j pre $j \in \{1, \ldots, n\}$ do určitých, približne rovnako veľkých disjunktných podmnožín a následného opakovania tohto procesu rozbíjania predstaveného v tejto časti práce.

V prvom kroku je množina Z rozbitá do určitých k približne rovnako veľkých disjunktných podmnožín B_i^1 pre $i \in \{1, \ldots, k\}$. Autori článku tieto podmnožiny nazývajú bloky [3]. Pre takto vytvorené bloky teda platí, že $\bigcup_{i=1}^{k} B_i^1 = Z$ a zároveň $B_i^1 \cap B_j^1 = \emptyset$ pre $i \neq j$.

Pre každú takto skonštruovanú podmnožinu B_i^1 pre $i \in \{1, \ldots, k\}$, je následne formulovaný podmodel (7) zodpovedajúci útvarom obsiahnutým v príslušnom bloku. Úlohy LP, ktoré budú riešené v rámci jednotlivých blokov budú mať rovnakú dimenziu ako pôvodná úloha, keďže počty ohraničení ktorým zodpovedá súčet rozmeru vstupov a výstupov $m+s$ ostáva nemenný, avšak kardinalita je podstatne nižšia.

Kardinalitu podúloh ovplyvňujeme kardinalitou bloku resp. počtom zvolených blokov. Autori článku [3] používajú parameter b, ktorý udáva maximálnu kardinalitu bloku v prvom kroku algoritmu. To znamená, že počet blokov k pri prvom rozbití úlohy je určený na základe zvolenej kardinality blokov, ktorú determinuje parameter b, teda platí $k = \lceil |Z|/b \rceil$. To má za následok, že sa nebude riešiť pôvodný model s kardinalitou n, čomu zodpovedá množina všetkých pozorovaných útvarov Z, ale istá sada k modelov s menšími počtami útvarov a teda nižšími kardinalitami.

Poznamenajme, že pri riešeniach úloh LP formulovaných pre jednotlivé bloky B_i^1 pre $i \in \{1, \ldots, k\}$, môžeme využiť metódu RBE uvedenú v predchádzajúcej podkapitole. Dodávame, že implementácia predchádzajúcej metódy má mať za cieľ synergický efekt vzhľadom k celkovému výpočtovému času.

Množinu všetkých efektívnych útvarov v rámci pôvodnej množiny Z budeme označovať E^*. Množinu všetkých neefektívnych útvarov v rámci pôvodnej množiny Z budeme označovať I^*. Množiny efektívnych, resp. neefektívnych útvarov jednotlivých blokov B_i^1 v rámci

prvého kroku rozbíjania budeme označovať ako $E(B_i^1)$, resp. $I(B_i^1)$, pričom $i \in \{1, \ldots, k\}$.

Podstata algoritmu je postavená tvrdení, ktoré je v [3] sformulované bez dôkazu ako Veta 2. Táto veta hovorí, že ak platí $B_i^1 \subseteq Z$, potom platí, že ak je útvar neefektívny v rámci niektorého z blokov B_i^1 pre $i \in \{1, \ldots, k\}$, potom musí byť nutne neefektívny aj v rámci množiny Z. To znamená, že platí $I(B_i^1) \subseteq I^*$. Zároveň podľa tej istej vety platí, že ak pozorovaný útvar je efektívny v rámci množiny Z, potom musí byť nutne efektívny v rámci niektorého z blokov B_i^1 pre $i \in \{1, \ldots, k\}$. To znamená, že platí $E^* \subseteq \bigcup_{i=1}^{k} E(B_i^1)$. Táto veta tvorí základ prezentovaného algoritmu, pričom poznamenajme, že je jednoduchým dôsledkom Tvrdenia 1.7, ktoré sme dokázali v prvej kapitole.

Výsledkom rozbitia základnej úlohy v prvom kroku na sadu podúloh a následnej aplikácií príslušných úloh LP, je obdržanie množín efektívnych, resp. neefektívnych útvarov v rámci jednotlivých blokov B_i^1 pre $i \in \{1, \ldots, k\}$. Neefektívne útvary v rámci jednotlivých blokov sú zjednotené, pričom zo spomenutej vety platí, že $\bigcup_{i=1}^{k} I(B_i^1) \subseteq I^*$. Efektívne útvary získane takýmto spôsobom sú následne zjednotené do jednej novej množiny $\bigcup_{i=1}^{k} E(B_i^1) = Z_U$, ktorá obsahuje všetky efektívne útvary, ktoré boli vyhodnotené vzhľadom k jednotlivým blokom. Pripomíname, že medzi takto získanými efektívnymi útvarmi sa určite nachádzajú, vychádzajúc zo spomínanej vety, všetky efektívne útvary vzhľadom k pôvodnej množine Z, keďže platí $E^* \subseteq \bigcup_{i=1}^{k} E(B_i^1) = Z_U$. Zároveň však dodávame, že množina Z_U môže obsahovať taktiež neefektívne útvary vzhľadom na pôvodnú množinu Z.

V druhom kroku algoritmu je získaná množina Z_U opäť rozbíjaná na k približne rovnako veľkých blokov, pričom veľkosť bloku je znova určená parametrom b. To znamená, že platí $k = \lceil |Z_U|/b \rceil$. Rozbitím množiny Z_U znovu vznikne systém blokov B_i^2 pre $i \in \{1, \ldots, k\}$, pričom platí $\bigcup_{i=1}^{k} B_i^2 = Z_U$ a zároveň $B_i^2 \cap B_j^2 = \emptyset$ pre $i \neq j$. Vyriešením úloh, prislúchajúcich blokom obdržíme opäť množiny efektívnych útvarov vzhľadom k príslušným blokom $E(B_i^2)$ pre $i \in \{1, \ldots, k\}$, pričom opäť platí, že $E^* \subseteq \bigcup_{i=1}^{k} E(B_i^2)$. Zároveň obdržíme množiny neefektívnych útvarov $I(B_i^2)$ pre $i \in \{1, \ldots, k\}$. Proces rozbíjania na bloky v druhom kroku algoritmu je ukončený za predpokladu, že podiel kardinality množiny $\bigcup_{i=1}^{k} E(B_i^2)$ a kardinality množiny Z_U, je väčší ako parameter γ.

Ak je podiel väčší ako zadaný parameter γ, potom pre útvary patriace množine $\bigcup_{i=1}^{k} E(B_i^2)$ je formulovaný nový model 7. Následkom jeho riešenia je extrakcia množiny

E^*, obsahujúca všetky efektívne útvary. Dodávame, že pri extrakcii množiny E^* môžeme využiť princíp RBE, predstavený v predošlej podkapitole.

Naopak, ak je spomínaný podiel vyšší ako parameter γ, potom množina $\bigcup_{i=1}^{k} E(B_i^2)$ je opäť rozbitá na ďalšiu sadu blokov. Dávame do pozornosti, že veľkosť blokov ako uvádzajú autori [3], bude zvyšovaná parametrom β. To znamená, že veľkosť blokov pri nasledovnom procese rozbíjania bude rovná βb. Následný postup prebieha analogicky.

Nie je ťažké si uvedomiť, že tento proces rozbíjania základnej úlohy na sadu úloh a následného zjednotenia efektívnych útvarov a opakovania tohto cyklu, bude mať za následok nutnosť vyššieho počtu riešení úloh LP. Tieto úlohy však budú s podstatne nižšími kardinalitami ako pôvodná úloha. Hlavnou výhodou tohto prístupu je možnosť určenia mnohých neefektívnych útvarov v rámci blokov, následkom čoho sú napriek zvýšenému počtu úloh LP signifikantné časové úspory.

Je však potrebné si uvedomiť, že ak pozorovaný útvar je v rámci niektorej podúlohy klasifikovaný ako neefektívny, je síce neefektívny aj v rámci celej skupiny útvarov, avšak optimálne riešenie pre takýto neefektívny útvar prislúchajúci niektorému bloku, nemôžeme nutne chápať ako výslednú mieru efektívnosti. Je tomu tak z toho dôvodu, že v danom bloku pre ktorý je formulovaný podmodel a ktorému prislúcha pozorovaný útvar, nemusia byť nutne obsiahnuté tie efektívne útvary, ktoré sa spolupodieľajú na jeho efektívnom vzore vzhľadom k pôvodnej úlohe. To znamená, že môže dochádzať k absenciám efektívnych útvarov, ktoré sa spolupodieľajú na evalvácii jeho miery efektívnosti, keďže optimálna hodnota zo submodelu nemusí nutne zachytávať celkovú mieru neefektívnosti.

Z tohto dôvodu sa algoritmus hierarchickej dekompozície delí na dve fázy. Hlavnú podstatu prvej fázy algoritmu sme popísali vyššie, pričom spočíva v klasifikácií všetkých efektívnych útvarov, resp. rozdelení pôvodnej sady dát na efektívne a neefektívne útvary.

Druhá fáza algoritmu hierarchickej dekompozície následne spočíva v aplikácii aditívneho modelu, nad množinou všetkých efektívnych útvarov, postupne na zistenie efektivity každého neefektívneho útvaru určeného vzhľadom k jednotlivým blokom. To znamená, že ľavé strany ohraničení v podmienkach prípustnosti aditívneho modelu 7 sú tvorené množinou všetkých efektívnych útvarov E^* zistených v prvej fáze, a zároveň pravé strany týchto ohraničení sú tvorené postupným dosádzaním útvarov vyhodnotených ako neefektívne v rámci jednotlivých blokov. Táto úloha má podstatne nižšiu kardinalitu ako pôvodná úloha,

pričom tá je podmienená práve hustotou pôvodnej úlohy, teda celkovým počtom efektívnych útvarov. Pre kompaktnejší popis algoritmu hierarchickej dekompozície uvádzame nižšie pseudokód.

HD pseudokód:

FÁZA 1:

Krok 1 $Z = \{DMU_1, \ldots, DMU_n\}$

$E^* \leftarrow \emptyset$

$I^* \leftarrow \emptyset$

$b \leftarrow$ kardinalita blokov v dekompozícii

$\beta \leftarrow$ parameter zväčšovania kardinality

$\gamma \leftarrow$ parameter kontroly

$l = 1$

$k_l = \lceil |Z|/b \rceil$-Voľba počtu blokov.

Riešenie úloh prislúchajúce blokom:

pre každý blok $B_1^l, B_1^l, \ldots, B_{k_l}^l$ riešenie modelu 7 s využitím princípu RBE, kde súčty v podmienkach prípustnosti berieme cez indexy j útvarov patriacich danému bloku.

$I^* = I^* \cup \bigcup_{i=1}^{k_l} I(B_i^l)$

$Z_U = \bigcup_{i=1}^{k_l} E(B_i^l)$

Krok 2 $l = l + 1$

$u = |Z_U|$

$k_l = \lceil |Z_U|/b \rceil$-Voľba počtu blokov.

Riešenie úloh prislúchajúce blokom:

pre každý blok $B_1^l, B_1^l, \ldots, B_{k_l}^l$ riešenie modelu 7 s využitím princípu RBE, kde súčty v podmienkach prípustnosti berieme cez indexy j útvarov patriacich danému bloku.

$I^* = I^* \cup \bigcup_{i=1}^{k_l} I(B_i^l)$

$Z_U = \bigcup_{i=1}^{k_l} E(B_i^l)$

Ak $|Z_U|/u > \gamma$, potom:

riešenie modelu 7, pre útvary obsiahnuté v Z_U s využitím princípu RBE, kde súčty v podmienkach prípustnosti berieme cez indexy j útvarov množine Z_U.

$E^* = E(Z_U)$

$I^* = I^* \cup I(Z_U)$

KONIEC 1.FÁZY.

Inak:

$b = \beta b$ a späť na začiatok Kroku 2.

FÁZA 2: Pre všetky útvary $DMU_o \in I^*$ riešime úlohu 7 kde súčty v podmienkach prípustnosti berieme cez indexy j útvarov patriacich do množiny E^*.

Barr a Durchholz uvádzajú, že veľkosť blokov ovplyvňuje celkový výpočtový čas [3]. Z tohto dôvodu je potrebné optimálnu počiatočnú veľkosť blokov b, pripadne optimálne hodnoty parametrov β, resp. γ empiricky vyhodnotiť. Keďže pre dáta s rôznymi hustotami, dimenziami a kardinalitami budú platiť rôzne optimálne hodnoty parametrov, potreba skúmania týchto hodnôt sa javí ako nevýhoda algoritmu. Je potrebné poznamenať, že napriek tomu, že optimálne hodnoty parametrov nie sú známe, algoritmus hierarchickej dekompozície je podstatne efektívnejší ako predchádzajúci spomenutý, čo bude demonštrované v tretej kapitole tejto práce. Je tomu tak z dôvodu, že hierarchická dekompozícia disponuje všetkými výhodami predchádzajúceho algoritmu, naopak to však neplatí.

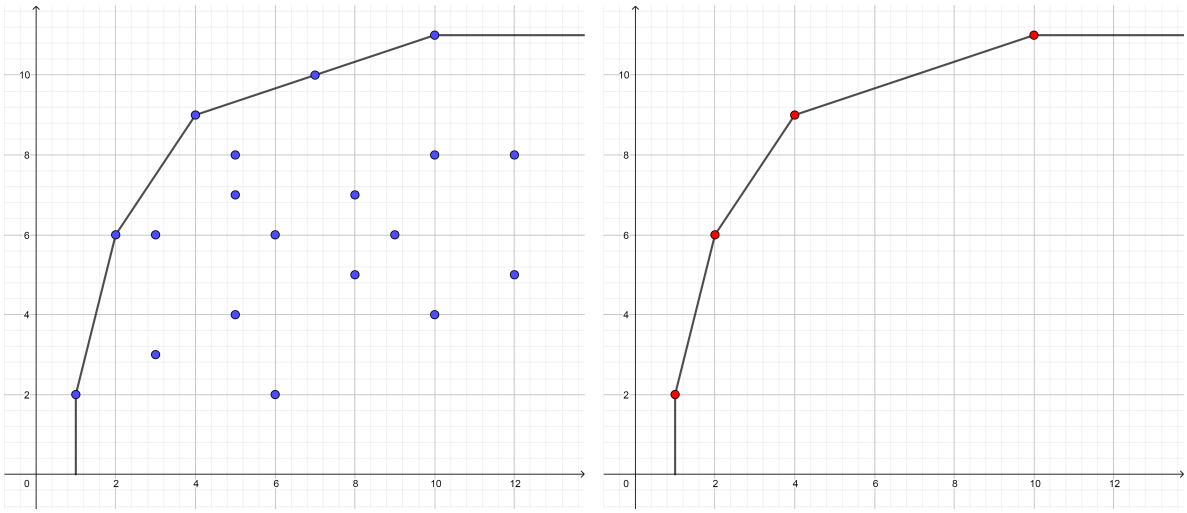
Aj keď algoritmus hierarchickej dekompozície je možné vykonať na jednom počítači, jeho ďalšia výhoda spočíva v možnosti paralelizácie spomínanej v úvode tejto kapitoly. Pre pripomenutie uvádzame, že proces paralelizácie znamená v istom zmysle rozloženie výpočtovej náročnosti pôvodného problému z jedného počítaču na viacero počítačov, prípadne z jedného jadra procesora na viacero jadier. K možnosti paralelizacie je potrebné splniť predpoklad nezávislosti výpočtov. Algoritmus hierarchickej dekompozície spĺňuje túto požiadavku. Je zrejme, že klasifikácia útvarov na efektívne, resp. neefektívne obsiahnuté v jednom bloku, nijakým spôsobom neovplyvňuje klasifikáciu útvarov obsiahnutých v inom bloku. Z tohto dôvodu je možne 1. fázu hierarchickú dekompozíciu vykonávať paralelne na viacerých jadrách procesora, resp. viacerých počítačoch. Dodávame, že taktiež 2. fázu hierarchickej dekompozície je možne vykonať paralelným spôsobom, keďže výpočty efektivít neefektívnych útvarov prebiehajú tradičným iteratívnym prístupom, ktorý je možné paralelizovať.

Výhodou algoritmu hierarchickej dekompozície je taktiež možnosť použitia odlišných

DEA modelov v 2. fáze, kedy je množina efektívnych útvarov známa, ktoré by viedli k iným hodnotám efektivít neefektívnych útvarov. Použitím aditívneho modelu v 1. fáze sme obdržali množinu efektívnych útvarov. Je zrejmé, že množina efektívnych útvarov by bola totožná aj za použitia iného modelu, keďže klasifikáciu útvaru na efektívny resp. neefektívny determinuje pozícia útvaru v množine produkčných možností a nie použitý model. Výhodou použitia aditívneho modelu oproti radiálnym modelom je, že nepotrebujeme druhú fázu na zistenie efektívnosti, keďže z optimálneho riešenia vieme jednoznačne určiť, či pozorovaný útvar je efektívny alebo nie. Niekedy nemusí byť zo začiatku zrejmé, ktorý DEA model je vhodné využiť. Práve v takýchto prípadoch sa môže javiť využitie hierarchickej dekompozície za vhodné, keďže pre iné DEA modely je potrebné už len riešenie 2. fázy.

2.2.3 Build Hull

Nasledujúcim algoritmom je algoritmus nazývaný *Build Hull*. Tento algoritmus bol predstavený roku 2011 jeho autorom J. H. Dulom [8]. Algoritmus pristupuje k riešeniu problému v dvoch fázach, rovnako ako v predchádzajúcom prípade hierarchickej dekompozície. V prvej fáze je identifikovaná tzv. *kostra* (angl. *frame*), ktorú budeme označovať symbolom K^*. Kostra K^* pozostáva z podmnožiny efektívnych útvarov, ktoré sú v zmysle terminológie DEA modelov označované ako superefektívne. Pripomíname, že pod superefektívnymi útvarmi rozumieme také efektívne útvary, ktorých odstránenie by spôsobilo zmenu množiny produkčných možností. Ako je uvedené v článku [8], kostra K^* je najmenšia možná podmnožina útvarov množiny Z generujúca totožné množiny produkčných možností. To znamená, že množina produkčných možností M generovaná množinou Z, je zhodná s množinou produkčných možností M generovanou množinou K^*. Dodávame, že rôzne výnosy z rozsahu vedú k rôznym množinám produkčných možností a teda nutne k rozdielnym kostrám. V tejto časti práce budeme uvažovať iba variabilné výnosy z rozsahu. Pre informácie o kostrách zodpovedajúce rozdielnym výnosom z rozsahu ako aj o ich funkciách, môže poslúžiť napr. publikácia [13]. Nižšie uvádzame Obr. 2 ktorý demonštruje, že množina produkčných možností M generovaná množinou Z, je totožná s množinou produkčných možností M generovanou kostrou K^*, ak uvažujeme rovnaké v našom prípade variabilné výnosy z rozsahu. Hlavný rozdiel algoritmu Build Hull oproti predošlým algo-



Obr. 2: Vľavo: Množina produkčných možností, generovaná Z.
Vpravo: Množina produkčných možností, generovaná K^*.
Zdroj: Vlastné spracovanie

ritmom spočíva v hľadaní množiny efektívnych útvarov potrebných k výpočtu mier efektívnosti zvyšných útvarov. Kým algoritmus hierarchickej dekompozície ako aj algoritmus RBE sa opierali o postupné odstraňovanie neefektívnych útvarov vedúcich k postupnému znižovaniu kardinality pôvodnej úlohy, J.H. Dula navrhol systém úloh postupnej identifikácie efektívnych útvarov tvoriacich kostru K^*. Množina zistených efektívnych útvarov patriacich kostre bude neskôr tvoriť vstupné údaje koeficientov vo výslednom modifikovanom aditívnom modeli, použitom v druhej fáze algoritmu Build Hull, na ohodnotenie zvyšných útvarov. Útvary nepatriace kostre pozostávajú z neefektívnych útvarov, prípadne tých efektívnych útvarov, ktoré ležia vo vnútrach niektorých stien, resp. vnútrach hrán polyédra patriacich hranici efektívnosti. Množinu útvarov, ktoré nepatria kostre K^*, budeme označovať NK^*. Dalo by sa povedať, že algoritmus využíva vo svoj prospech podstatne viac z teórie lineárneho programovania ako predošlé dva spomenuté algoritmy. Prvá fáza algoritmu Build Hull korešponduje s geometrickým problémom nájdenia vrcholov neohraničeného konvexného polyédra, ktorý zodpovedá množine produkčných možností M. V následnom uvedieme definíciu krajného bodu polyédra.

Definícia 2.1. *Nech $S \subseteq \mathbb{R}^n$ je ľubovoľná konvexná množina. Bod $s \in S$ nazveme krajným bodom množiny S, ak neexistujú body $x, y \in S$ a číslo $\lambda \in (0,1)$ také, že $x \neq y$ a $s = \lambda x + (1-\lambda)y$.*

Ako sme uviedli v prvej kapitole tejto diplomovej práce, množina produkčných možností M je popísaná systémom ohraničení zodpovedajúcich podmienkam prípustnosti. Táto sústava podmienok vytvára neohraničený uzavretý polyéder. Berúc do úvahy, že pre potreby ohodnotenia neefektívnych útvarov ako aj efektívnych, ktoré sú vo vnútrach stien, resp. hrán polyédra potrebujeme poznať iba hranicu efektívnosti tohto polyédra, je postačujúce pri jej určení poznať iba krajné body polyédra, keďže hranica efektívnosti je tvorená určitými konvexnými kombináciami jeho krajných bodov.

Prvá fáza, ako sme spomenuli vyššie, je založená na nájdení efektívnych útvarov prislúchajúcich kostre K^*, ktorá generuje množinu produkčných možností M. Tento proces je vykonávaný iteratívne, čo znamená, že v každej iterácii je odhalený minimálne jeden prvok patriaci kostre K^*, pričom každá iterácia pozostáva z riešenia systému úloh, ktoré J.H. Dula v článku [8] navrhol.

Podmnožinu množiny K^*, budeme označovať K. Je zrejmé, že ak $K \subset K^*$, tak množina produkčných možností M, ktorá je generovaná množinou K, je podmnožinou množiny produkčných možností M, generovanej pomocou K^*. Množinu M, generovanú pomocou množiny K budeme označovať ako parciálnu. Je zrejmé, že parciálna množina produkčných možností M, generovaná množinou K, sa postupnými iteráciami zväčšuje, keďže následkom pridávania superefektívnych útvarov sa zväčšuje množina K.

V článku [8] autor uvádza, že počiatočná parciálna množina produkčných možností je generovaná niektorým z prvkov množiny K^*. Zároveň však dodáva, že algoritmus bude fungovať pre parciálnu množinu produkčných možností generovanú ľubovoľným inicializačný bodom.

Prvá zo systému úloh vedúca k určeniu ďalšieho superefektívneho útvaru množiny produkčných možností M, generovanej množinou všetkých pozorovaných útvarov Z, teda prvku kostry K^*, je formulovaná pre potreby určenia, či pozorovaný útvar patrí do parciálnej množiny produkčných možností M, generovanej množinou K. Formuláciu tejto optimalizačnej úlohy, ktorá je uvedená nižšie, čerpáme z článku [8].

$$
\begin{aligned}
\alpha^* = \min_{\alpha,\lambda} \ & \alpha \\
s.t. \ & \sum_{j \in K} x_j \lambda_j - \alpha e_m \leq x_o, \\
& \sum_{j \in K} y_j \lambda_j + \alpha e_s \geq y_o, \\
& \sum_{j \in K} \lambda_j = 1, \\
& \lambda \geq 0, \alpha \geq 0
\end{aligned} \tag{14}
$$

Vyššie uvedená úloha, ktorú označuje J. Dula ako primárnu úlohu, je vždy prípustná a ohraničená, pričom dôkaz tohto tvrdenia je uvedený v [8]. Výsledkom riešenia tejto úlohy je, či sa testovaný útvar (x_o, y_o), nachádza mimo parciálnej množiny produkčných možností M, ktorá je generovaná množinou K.

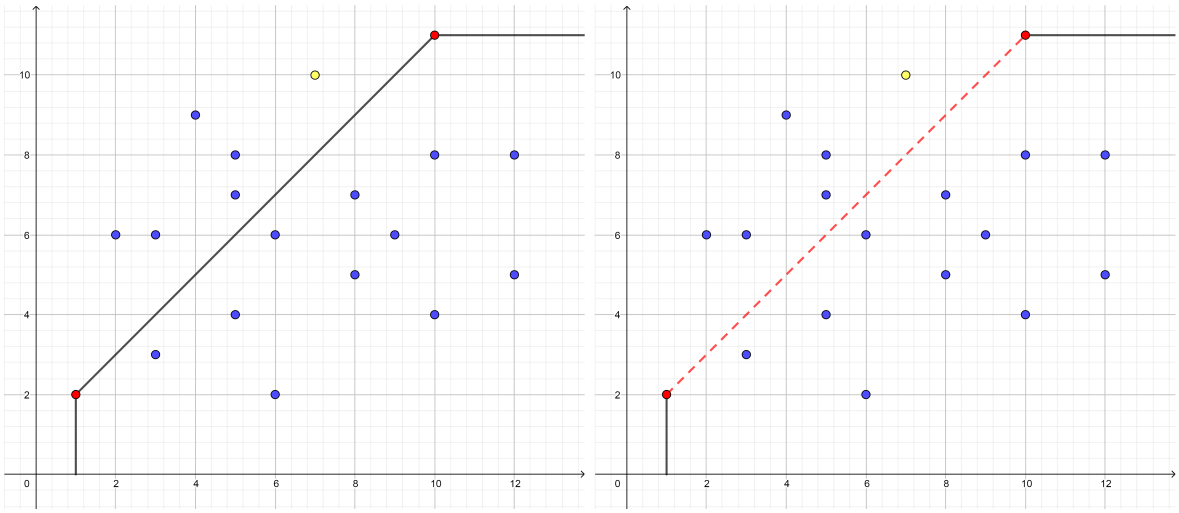
Vystupujúce vektory e_m, resp. e_s v primárnej úlohe sú vektory samých jednotiek príslušného rozmeru, teda m resp. s. Vektor $(e_m, -e_s)$ predstavuje smer posunu bodu (x_o, y_o). Optimálna hodnota α v modeli 14, predstavuje koeficient najmenšieho posunu bodu (x_o, y_o) v danom smere $(e_m, -e_s)$, ktorý zabezpečí, že posunutý bod už bude z množiny M generovanej pomocou K. Nie je ťažké si uvedomiť, že ak α^* vyjde kladné, potom z konštrukcie úlohy je zrejmé, že pozorovaný útvar DMU_o, ktorému zodpovedá prislúchajúca dvojica (x_o, y_o), nepatrí do parciálnej množiny produkčných možností M, generovanej množinou K. Naopak, ak $\alpha^* = 0$, potom pozorovaný útvar DMU_o je z parciálnej množiny produkčných možností M a môžeme ho zaradiť do množiny útvarov nepatriacich kostre NK^*.

Ak pre pozorovaný útvar (x_o, y_o) vyšlo jeho $\alpha^* > 0$ a daný útvar sa tak nachádza mimo parciálnej množiny produkčných možností M generovanej množinou K, potom, na základe teórie lineárneho programovania, vieme, že existuje nadrovina, ktorá umožňuje útvar (x_o, y_o) ostro separovať od parciálnej množiny produkčných možností M, keďže M je konvexná. Ak však uvážime neostrú separáciu, potom je možné pozorovaný útvar (x_o, y_o) separovať od parciálnej množiny produkčných možností M nadrovinou, ktorá prechádza jednou z jej stien. To znamená, že sa jedná o opornú nadrovinu parciálnej množiny produkčných možností M, generovanej pomocou množiny K. Odpoveď na predpis takejto opornej nadroviny dáva riešenie úlohy uvedenej nižšie, ktorej formuláciu čerpáme z [8], a

ktorú autor označuje ako duálnu.

$$\begin{aligned} \alpha^* = \max_{u,v,\sigma} \; & u^T y_o - v^T x_o + \sigma \\ s.t. \; & u^T y_j - v^T x_j + \sigma \leq 0, \forall (x_j, y_j) \in K \\ & u^T e_s + v^T e_m \leq 1 \\ & u \geq 0, v \geq 0 \end{aligned} \tag{15}$$

Výsledkom vyššie formulovanej duálnej úlohy je optimálne riešenie, ktorým dostávame opornú nadrovinu parciálnej množiny produkčných možností M. Táto oporná nadrovina, ktorá je daná rovnicou $u^{*T} y - v^{*T} x + \sigma^* = 0$, separuje parciálnu množinu produkčných možností M od útvaru (x_o, y_o), ktorý nepatrí danej množine [8]. Z teórie duality v LP je zrejmé, že ak primárna úloha bola prípustná a ohraničená, potom je aj jej duálny náprotivok prípustný a ohraničený a hodnoty účelových funkcií sa rovnajú. Dôkaz tohto tvrdenia je uvedený napr. v [19]. Nižšie uvádzame Obr. 3, ktorý znázorňuje separujúcu opornú nadrovinu parciálnej množiny produkčných možností M, generovanej množinou K, ktorej predpis dostávame z optimálneho riešenia duálnej úlohy. Pripomíname, že pred prvou iteráciou množina K pozostáva iba z takých útvarov množiny Z, ktoré majú niektorú hodnotu výstupu maximálnu, resp. niektorú hodnotu vstupu minimálnu a sú jediné.



Obr. 3: Vľavo: Parciálna produkčná množina, generovaná dvoma červenými bodmi. Vpravo: Čiarkovane je vyznačená časť opornej nadroviny separujúcej žltý bod od parciálnej produkčnej množiny.

Zdroj: Vlastné spracovanie

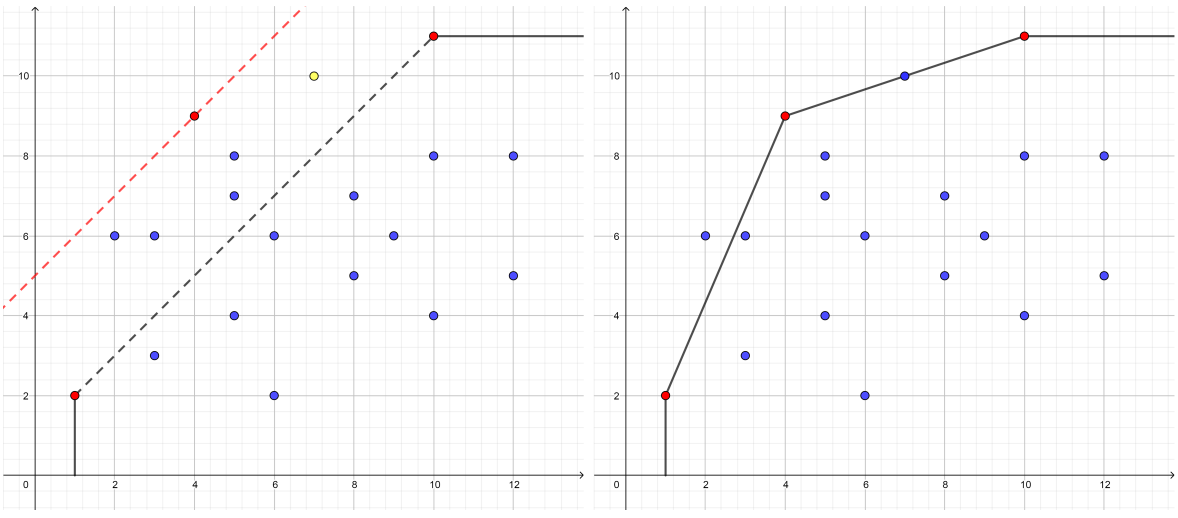
Pre úplnosť mechanizmu nájdenia nového útvaru patriaceho kostre K^* je potrebné rie-

šenie ešte poslednej úlohy, ktorá zabezpečuje maximálny paralelný posun získanej opornej nadroviny $u^{*T}y - v^{*T}x + \sigma^* = 0$. To znamená, že takto posunutá oporná nadrovina bude opornou nadrovinou množiny produkčných možností M generovanej množinou všetkých útvarov Z. Úlohu, ktorú uvádzame nižšie, čerpáme z [8].

$$DMU^* = (x^*, y^*) = \underset{(x,y)\in Z}{\operatorname{argmax}}\, u^{*T}y - v^{*T}x + \sigma^* \tag{16}$$

Ak riešením vyššie uvedenej úlohy je jediný útvar, potom daný útvar, podľa [8], je prvkom kostry K^*. Následne je takto získaný útvar pridaný do množiny K. V prípade, že riešením úlohy je viacero útvarov, potom tieto útvary generujú určitú množinu produkčných možností. Dané útvary sú testované, či sú superefektívne vzhľadom k množine produkčných možností, ktorú generujú. Ak sú superefektívne vzhľadom k množine produkčných možností, ktorú generujú, potom sú prvkami kostry [8]. Následne sú všetky takto získané útvary vložené do množiny K.

Je zrejmé, že v nasledovnej iterácii bude množina K väčšia a teda bude nutne väčšia taktiež parciálna množina produkčných možností M ňou tvorená. Pre lepšie pochopenie uvádzame nižšie Obr. 4, ktorý zobrazuje maximálny paralelný posun nadroviny ako aj pridanie útvaru do množiny K, čoho následkom je zmena parciálnej množiny produkčných možností oproti prvej iterácii.



Obr. 4: Vľavo: Maximálny posun opornej nadroviny.
Vpravo: Nová parciálna množina produkčných možností.
Zdroj: Vlastné spracovanie

Predchádzajúci postup sa znovu opakuje, pokým nie sú klasifikované všetky útvary

Z do množín K, resp. NK^*. Akonáhle sú všetky útvary rozdelené na útvary kostry K, resp. útvary nepatriace kostre NK^*, je potreba evalvácie mier efektívnosti útvarov nepatriacich kostre NK^*, čomu zodpovedá druhá fáza algoritmu Build Hull. Nižšie uvádzame pseudokód algoritmu Build Hull.

Build Hull pseudokód:

FÁZA 1:

Krok 1 Inicializácia.

$Z = \{DMU_1, \ldots, DMU_n\}$

$NK^* \leftarrow \emptyset$

$K \leftarrow$ lubovoľný prvok množiny Z, ktorý je superefektívny.

$Z \leftarrow Z \setminus K$

Krok 2 Iterácie.

Riešenie úloh (14)/(15) pre $DMU_o \in Z$, kde súčty v podmienkach prípustnosti sa berú cez indexy j útvarov aktuálnej množiny K.

Ak $\alpha^* > 0$, potom:

pre (u^*, v^*, σ^*) riešenie úlohy 16.

Ak je riesením jeden útvar DMU^*, potom:

$K = K \cup \{DMU^*\} \quad \wedge \quad Z = Z \setminus \{DMU^*\}$.

Ak je riešením viacero útvarov, potom:

zistiť, ktoré sú superefektívne vzhľadom k množine produkčných možností ktorú generujú. Tie, ktoré sú superefektívne, vložiť do K a odstrániť zo Z.

Ak $\alpha^* = 0$, potom:

$NK^* = NK^* \cup \{DMU_o\} \quad \wedge \quad Z = Z \setminus \{DMU_o\}$

Krok 3 Kontrola.

Ak $Z \neq \emptyset$, potom:

späť na Krok 2,

inak:

koniec FÁZY 1.

FÁZA 2: Riešenie úlohy 7 pre $DMU_o \in NK^*$, kde súčty v podmienkach prípustnosti sa berú cez indexy j útvarov z množiny K.

Poznamenajme, že na základe Tvrdenia 1.6 vieme, že útvary v množine Z, ktoré majú maximálnu hodnotu niektorého z výstupov, resp. minimálnu hodnotu niektorého z vstupov, patria hranici množiny M, generovanej množinou Z. Na základe článku [12], ktorý sa venuje algoritmom odhaľujúcim efektívne útvary bez potreby riešenia úlohy LP, sú takto získané útvary jednoznačne superefektívne, pokiaľ pre každý vstup, resp. výstup existuje takýto útvar práve jeden. To znamená, že týmto spôsobom vieme získať najviac $m+s$ útvarov, ktoré budú jednoznačne prvkami kostry K^*. V nami naprogramovanom algoritme budú takto získané útvary tvoriť inicializačnú množinu K, ktorá generuje inicializačnú parciálnu množinu produkčných možností.

Vyššie popísaný mechanizmus pozostávajúci zo série úloh zaručuje získanie minimálnej množiny útvarov, ktoré sú postačujúce k výpočtu mier efektívnosti zvyšných útvarov. Táto konštrukcia nám dáva mnohé výhody. Keďže útvary obsiahnuté v kostre K^* sú jednoznačne efektívne, ich príslušné miery efektívností sú známe a stačí získať miery efektívností zvyšných útvarov. Akýkoľvek model sa rozhodneme využiť v druhej fáze, stačí pri hľadaní optimálnej hodnoty príslušnej úlohy prechádzať v podmienkach prípustnosti množinou K^*. To nám dáva rovnaké výhody, ako v prípade metódy hierarchickej dekompozície.

2.2.4 Návrh nového algoritmu

Posledným algoritmom predstaveným v tejto kapitole je nami navrhnutý algoritmus. Rovnako ako v prípade algoritmu hierarchickej dekompozície, resp. algoritmu Build Hull, aj tento algoritmus bude pristupovať k riešeniu problému v dvoch fázach. To znamená, že účelom prvej fázy bude zistiť množinu takých efektívnych útvarov, ktorú budeme značiť E^*, a ktorá bude postačujúca na výpočet mier efektívnosti neefektívnych útvarov, prípadne efektívnych útvarov ležiacich vo vnútrach niektorých stien, resp. hrán hranice efektívnosti H_E, množiny produkčných možností M, generovanej pomocou Z. Množinu útvarov nepatriacich množine E^* budeme označovať symbolom NE^*. Získané efektívne

útvary obsiahnuté v množine E^* budú neskôr spoluvytvárať efektívne vzory pri aplikácii aditívneho modelu v druhej fáze algoritmu. Náš prístup podobne ako algoritmus Build Hull, využíva zavedený pojem parciálnej množiny produkčných možností. Pripomíname, že sa jedná o takú množinu produkčných možností M, ktorá je generovaná vzhľadom na neúplnú množinu Z. Dodávame, že v tejto časti práce budeme uvažovať iba variabilné výnosy z rozsahu.

Nami navrhnutý algoritmus začína inicializačnou parciálnou množinou produkčných možností M, generovanej vzhľadom k množine KE, v ktorej sa budú nachádzať tzv. kandidáti na efektívne útvary. V inicializačnej fáze bude množina KE obsahovať všetky útvary s minimálnou hodnotou niektorého vstupu, resp. maximálnou hodnotou niektorého výstupu spomedzi všetkých útvarov patriacich množine Z. Je zrejmé, že ak existuje viacero útvarov s maximálnou hodnotou niektorého výstupu, resp. minimálnou hodnotou niektorého vstupu, potom sa medzi takto získanými útvarmi môžu objaviť aj nejaké útvary ležiace na hranici pseudoefektívnosti parciálnej množiny produkčných možností M generovanej vzhľadom ku množine KE. Následne sú z množiny Z odstránené všetky útvary patriace množine KE.

V prvej iterácii riešime, či útvar $(x_o, y_o) \in Z$ patrí parciálnej množine produkčných možností M, generovanej množinou KE. K riešeniu tejto úlohy využívame aditívny model pre výpočet superefektivity (10) predstavený v prvej kapitole. Dodávame, že ľavé strany ohraničení v podmienkach prípustnosti sú tvorené útvarmi obsiahnutými v množine KE a pravé strany zodpovedajú pozorovanému útvaru (x_o, y_o). Ako bolo uvedené v Kapitole 1, ak optimálna hodnota účelovej funkcie príslušného modelu pre pozorovaný útvar (x_o, y_o) vyjde rovná nule, teda $A_O^{SE*} = 0$, potom daný útvar patrí parciálnej množine produkčných možností M, generovanej množinou KE. Útvar (x_o, y_o) je následne odstránený z množiny Z a pridaný do množiny NE^*. Ak optimálna hodnota účelovej funkcie vyjde kladná, teda $A_O^{SE*} > 0$, potom útvar (x_o, y_o) je superefektívny vzhľadom k množine produkčných možností M, generovanej pomocou KE. Takto vyhodnotený útvar je následne odstránený z množiny Z a vložený do množiny KE. Dodávame, že pokiaľ (x_o, y_o) je superefektívny vzhľadom k množine KE, nemusí byť nutne superefektívny vzhľadom k množine Z. Je však zrejmé, že množina KE, do ktorej bol pridaný útvar (x_o, y_o), vytvára novú aproximáciu množiny produkčných možností M.

Vyššie uvedený postup opakujeme pokým nie je množina Z prázdna, teda pokým nie sú všetky útvary rozdelené do množín KE a NE^*. Je zrejmé že postupom uvedeným vyššie obdržíme množinu útvarov KE, pričom z predstavenej konštrukcie je možné, že v množine KE sa vyskytujú aj niektoré neefektívne útvary. To znamená, že o útvaroch v množine KE nevieme bez dodatočného riešenia rozhodnúť, ktoré útvary sú efektívne a ktoré nie. Z tohto dôvodu je potrebné nasledovné riešenie. Na získanú množinu KE aplikujeme obyčajný aditívny model (7), čoho dôsledkom je extrakcia množiny efektívnych útvarov E^*, a taktiež obdržanie mier efektívností neefektívnych útvarov množiny KE. Dávame do pozornosti, že pre zrýchlenú extrakciu efektívnych útvarov E^* nevyužívame tradičný prístup, ale algoritmus RBE predstavený v podkapitole 2.2.1, ktorý vedie k ďalším časovým úsporám.

Následkom vyššie popísaného postupu je obdržanie množiny efektívnych útvarov E^*, ktoré následne použijeme v príslušnom modeli na ohodnotenie zvyšných útvarov patriacich množine NE^* v druhej fáze výpočtu. Dávame do pozornosti, že druhá fáza môže byť vykonaná, tak ako aj v prípade algoritmu hierarchickej dekompozície ako aj algoritmu Build Hull, paralelne na viacerých zariadeniach súčasne. Konštrukcia nami predstaveného algoritmu môže v istých prípadoch byť účinnejšia ako algoritmus Build Hull ako budeme demonštrovať v nasledujúcej kapitole. Pre komplexnejší zápis algoritmu uvádzame nižšie pseudokód.

Pseudokód:

FÁZA 1:

Krok 1 Inicializácia.

$Z \leftarrow \{DMU_1, \ldots, DMU_n\}$

$KE \leftarrow$ Útvary s maximálnou hodnotou niektorého výstupu, resp. minimálnou hodnotou niektorého vstupu

$Z = Z \setminus KE$

$E^* \leftarrow \emptyset$

$NE^* \leftarrow \emptyset$

Krok 2 Riešenie úlohy (10) pre útvar $DMU_o \in Z$, kde súčty v podmienkach prípustnosti

berieme cez indexy j útvarov patriacich do množiny KE. Optimálnu hodnotu účelovej funkcie označíme A_O^{SE*}.

Krok 3 Aktualizácia.

$Z = Z \setminus \{DMU_o\}$

Ak $A_O^{SE*} = 0$, potom:

$NE^* = NE^* \cup \{DMU_o\}$

Ak $A_O^{SE*} > 0$, potom:

$KE = KE \cup \{DMU_o\}$

Krok 4 Kontrola.

Ak $Z \neq \emptyset$:

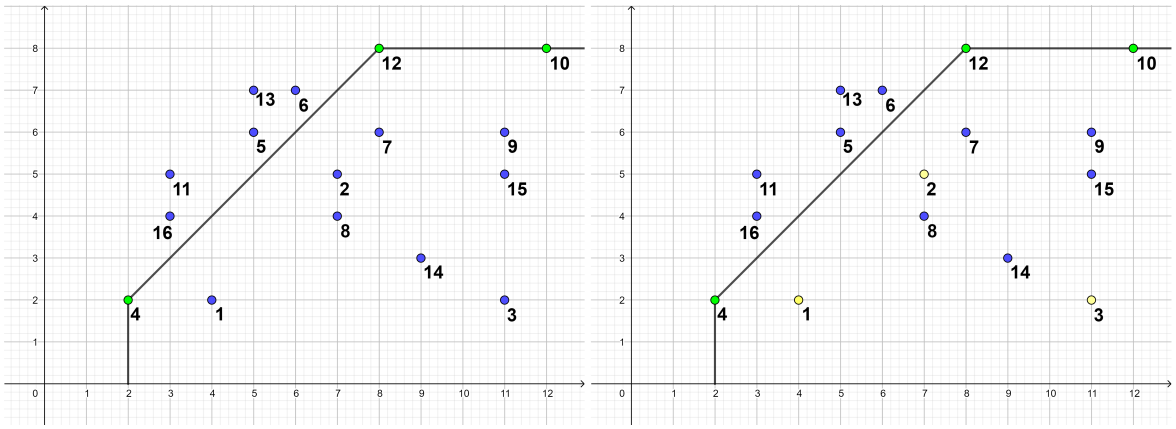
späť na Krok 2.

Inak:

Riešenie modelu 7, postupne pre útvary $DMU_o \in KE$ s využitím algoritmu RBE.

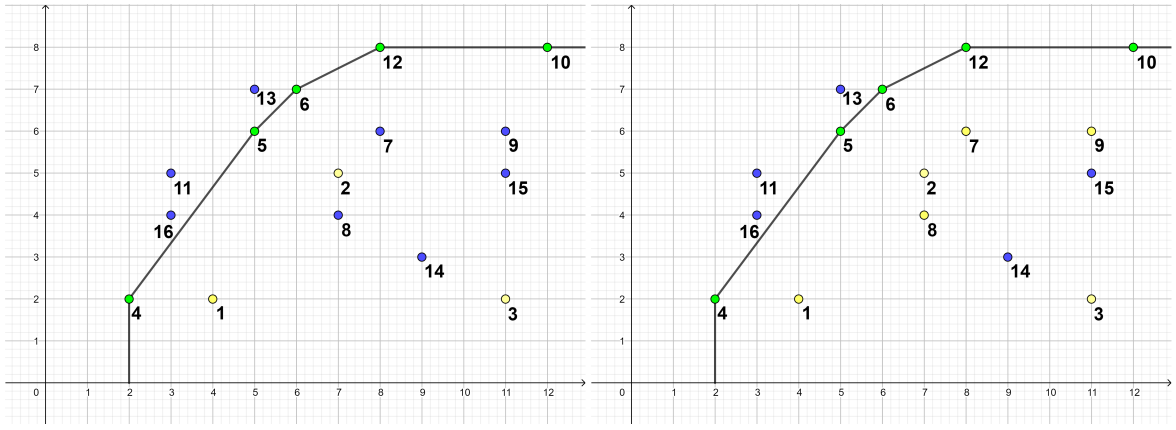
FÁZA 2: Pre všetky útvary $DMU_o \in NE^*$ riešenie úlohy (7), kde súčty v podmienkach prípustnosti berieme cez indexy j útvarov patriacich do množiny E^*.

Nižšie demonštrujeme sériu obrázkov pre lepšiu predstavu fungovania nami prezentovaného algoritmu. V prezentovanom príklade pozostáva množina pozorovaní zo 16 útvarov označených príslušnými indexami. Ako sme v predchádzajúcom uviedli, prvá parciálna množina produkčných možností je tvorená útvarmi, ktoré majú maximálnu hodnotu výstupu (útvary 10, 12) a minimálnu hodnotu vstupu (útvar 4), pričom tieto útvary sú následne vložené do KE a odstránené z množiny Z. Následne, pre zvyšné útvary množiny Z, testujeme pomocou modelu (10), či patria parciálnej produkčnej množine zobrazenej na Obr. 5 vľavo, pričom súčty v podmienkach prípustnosti berieme cez indexy j útvarov patriacich do množiny KE. Útvary 1, 2, 3, označené žltou farbou patria aktuálnej parciálnej množine produkčných možností, a teda sú vložené do množiny NE^*. Tieto útvary sú na Obr. 5 vpravo vyznačené žltou farbou.



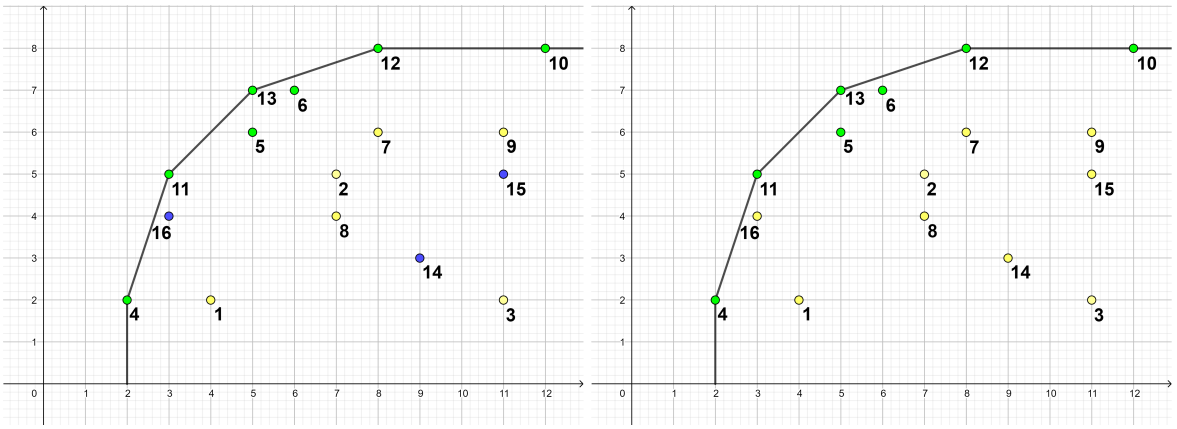
Obr. 5: Postup navrhovaného algoritmu 1

Útvar 5 sa nachádza mimo aktuálnej parciálnej množiny produkčných možností, takže je vložený do množiny KE, čo povedie k novej parciálnej produkčnej množine. Útvar 6 sa takisto nachádza mimo, čiže je taktiež zaradený do množiny KE, čo znova povedie k novej parciálnej produkčnej množine, ktorá je demonštrovaná na nasledovnom Obr. 6 vľavo. Útvary 7, 8, 9, označené žltou farbou, sa nachádzajú vo vnútri aktuálnej parciálnej množiny produkčných možností, čiže sú zaradené do NE^*, čomu zodpovedá pravá časť obrázku Obr. 6.



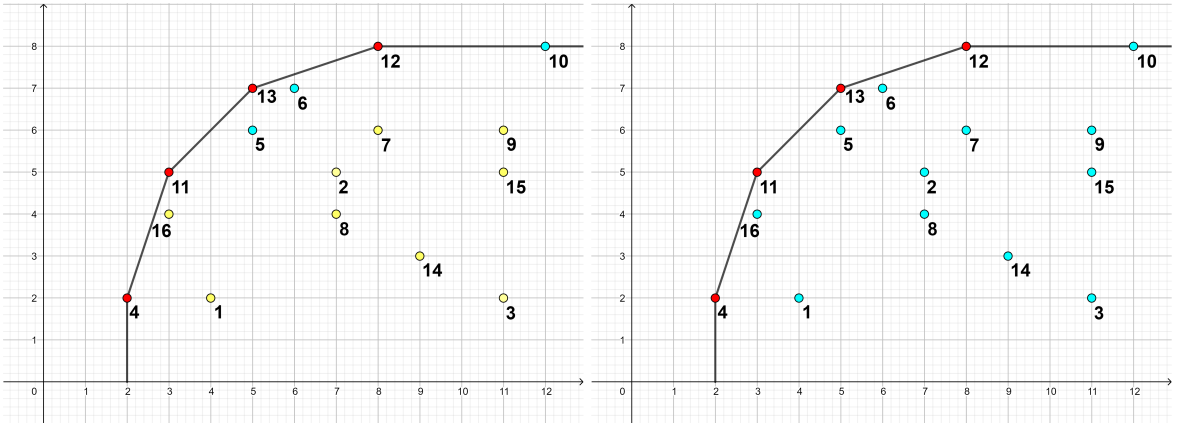
Obr. 6: Postup navrhovaného algoritmu 2

Útvary 11, 13 sa nachádzajú mimo, čiže sú zaradené do množiny KE, čo povedie znovu k zmene parciálnej množiny produkčných možností, čo môžeme vidieť na Obr. 7 vľavo. Útvary 14, 15, 16 sa nachádzajú vnútri produkčnej množine, čiže sú zaradené do množiny NE^*. Na Obr. 7 vpravo je vidieť rozdelenie útvarov na zelené, patriace množine KE^*, a žlté, patriace množine NE^*.



Obr. 7: Postup navrhovaného algoritmu 3

Zelené útvary sú kandidáti na efektívnych KE, pričom sa medzi nimi nachádzajú aj niektoré neefektívne útvary. Pre tieto útvary je vyriešený klasický aditívny model (7) s využitím algoritmu RBE. Vyriešením príslušných úloh LP obdržíme efektívne útvary množiny KE, pričom sa jedná zároveň o efektívne útvary vzhľadom k pôvodnej množine Z. Tieto útvary sú vidieť na Obr. 8 vľavo ako červené. Zároveň sme obdržali miery efektívností neefektívnych útvarov patriacich do množiny KE^* označených na Obr. 8 vľavo tyrkysovou farbou. Na záver sú dopočítané efektivity útvarov patriacich množine NE, ktoré sú označené na Obr. 8 vľavo žltou farbou, kde je báza tvorená efektívnymi útvarmi, ktoré sú označené červenou farbou.



Obr. 8: Postup navrhovaného algoritmu 4

3 Numerické experimenty

V poslednej kapitole tejto práce sa venujeme empirickému porovnávaniu jednotlivých algoritmov, predstavených v predchádzajúcich častiach, na dátových súboroch rôznych kardinalít, dimenzii a hustôt, pričom sa primárne zameriavame na rýchlosť výpočtov. Výsledky, ktoré prezentujeme v nasledujúcej časti, sme obdržali vlastnými výpočtami v softvéri MATLAB, v ktorom sme naprogramovali dané metódy podľa postupov uvedených v druhej kapitole. Podotýkame, že pri riešeniach jednotlivých úloh LP sme využívali metódu vnútorného bodu a zároveň dodávame, že rýchlosť výpočtov jednotlivých algoritmov boli merané pomocou matlabovských funkcii *tic* a *toc*. Dáta na ktorých boli algoritmy testované, sme čerpali z internetového zdroja [9].

3.1 Numerické výsledky

Ako prvé budeme vykonávať analýzu na simulovaných dátových sadách stredných kardinalít, postupne pre 5000, 7500 a 10000 pozorovaní, pričom dáta každej kardinality, sú zastúpené dimenziami 5 (2 vstupy, 3 výstupy), 10 (4 vstupy, 6 výstupov), 15 (7 vstupov, 8 výstupov) a 20 (9 vstupov, 11 výstupov) a zároveň sú všetky dátové sety zastúpené hustotami 1%, 13% a 25%. Dodávame, že dátové sady uvedené v jednotlivých tabuľkách, označujeme systémom kardinalita-dimenzia-hustota.

Nasledovná Tabuľka 1 poskytuje prehľad rýchlostí jednotlivých algoritmov, pričom sa jedná o súhrnný čas potrebný pre výpočet efektivít všetkých útvarov zadanej dátovej sady. Podotýkame, že pre jednotlivé dátové súbory, za použitia rozdielnych metód, vyšla totožná množina efektívnych útvarov a taktiež rovnaké hodnoty efektivít neefektívnych útvarov. Z prezentovaných výsledkov môžeme vidieť, že celkové výpočtové časy zväčšovaním dimenzie alebo kardinality dát narastajú použitím ľubovoľného algoritmu. Kým však celkový čas potrebný pre výpočet tradičným prístupom na dátovej sade 10000-20-01 narástol v porovnaní s dátovou sadou 5000-05-01 približne 18-násobne, použitím nášho prístupu narástol celkový čas len zhruba 3-násobne. To značí nárast efektivity nášho prístupu oproti tradičnému prístupu, pri náraste kardinality príp. dimenzie. Platí to taktiež pre metódu hierarchickej dekompozície ako aj Build Hull. Dodávame, že z prezentovaných výsledkov, pre dané dátové sady, obstál najefektívnejšie nami navrhnutý algoritmus, avšak

použitím ľubovoľnej metódy sme obdržali viditeľné časové úspory.

dátový súbor	tradičný	RBE	HD-RBE	BH	nový
5000-5-01	104.53	63.33	55.50	41.45	39.25
5000-10-01	133.80	80.52	60.63	46.22	44.36
5000-15-01	197.00	111.65	72.86	48.82	47.29
5000-20-01	364.63	174.84	90.92	54.31	52.23
7500-5-01	249.97	134.91	84.93	69.69	66.37
7500-10-01	326.66	189.35	97.25	74.73	72.20
7500-15-01	527.89	265.6	118.28	81.20	77.70
7500-20-01	1020.32	450.94	131.83	84.55	80.39
10000-5-01	419.94	227.10	108.90	88.52	83.36
10000-10-01	574.96	313.75	132.95	103.47	99.23
10000-15-01	1196.82	493.43	159.19	115.40	109.77
10000-20-01	1806.83	844.61	193,65	128.74	119.76

Tabuľka 1: Informácie o výsledkoch prislúchajúcich dátovým sadám s hustotami 1% pre jednotlivé algoritmy

Z výsledkov uvedených v Prílohe A v Tabuľke 7 vyplýva, že časové úspory použitím metódy RBE, oproti použitiu tradičného prístupu výpočtu, sa nachádzajú v rozmedzí 39.41% - 58.77%. Časové úspory použitím metódy hierarchickej dekompozície sa pohybujú v rozmedzí 46.91% - 89.28%, úspory dosiahnuté využitím metódy Build Hull sú v rozmedzí 60.35% - 92.87% a napokon úspory využitím nášho algoritmu sa nachádzajú v rozmedzí 62.45% - 93.37%, oproti celkovému výpočtovému času za použitia tradičného prístupu výpočtu. Z prezentovaných výsledkov, v prípade hierarchickej dekompozície, Build Hull a nášho algoritmu, je badateľná taktiež kladná závislosť medzi rozmerom dátovej sady a efektívnosťou uvedených metód.

Tabuľka 2, uvedená nižšie, poskytuje prehľad rýchlostí jednotlivých algoritmov, pre dátové sady rovnakých kardinalít a dimenzii ako v predošlej Tabuľke 1, avšak s rozdielnou hustotou každej dátovej sady, ktorá je na úrovni 13%. Tak ako v predchádzajúcej tabuľke sa jedná o súhrnný čas potrebný pre výpočet, pričom opäť podotýkame, že pre jednotlivé dátové súbory, za použitia rozdielnych metód, vyšla totožná množina efektívnych útvarov a taktiež rovnaké hodnoty efektivít neefektívnych útvarov. Oproti predchádzajúcim výsledkom môžeme pre každý dátový súbor pozorovať nárast celkového výpočtového času, čo je zapríčinené nárastom hustoty v dátovej sade.

dátový súbor	tradičný	RBE	HD-RBE	BH	nový
5000-5-13	132.49	81.57	71.05	58.97	58.39
5000-10-13	188.99	111.94	83.93	70.22	68.40
5000-15-13	282.94	168.98	106.38	92.94	91.03
5000-20-13	299.39	190.48	118.94	101.26	99.92
7500-5-13	298.02	165.18	112.35	102.66	101.15
7500-10-13	397.59	239.73	139.48	129.92	124.92
7500-15-13	641.38	332.82	166.74	156.15	152.90
7500-20-13	821.94	449.77	203.19	194.33	190.57
10000-5-13	555.04	319.04	185.85	175.62	174.22
10000-10-13	715.92	426.53	205.65	199.44	194.57
10000-15-13	1220.46	664.34	255.19	254.75	250.03
10000-20-13	1702.61	957.57	316.87	320.49	313.75

Tabuľka 2: Informácie o výsledkoch prislúchajúcich dátovým sadám s hustotami 13% pre jednotlivé algoritmy

Tabuľka 8 uvedená v prílohe ukazuje, že časové úspory RBE prístupu oproti tradičnému sa nachádzajú v rozmedzí 36.38% - 48.11%. Časové úspory metódy hierarchickej dekompozície sa pohybujú v rozmedzí 46.37% - 81.39%, úspory použitím metódy Build Hull sú v rozmedzí 55.49% - 81.18% a napokon úspory nášho algoritmu sa nachádzajú v rozmedzí 55.93% - 81.57%, oproti tradičnému prístupu výpočtu. Z prezentovaných výsledkov je zrejmé, že účinnosti jednotlivých algoritmov dosahujú nižšie hodnoty oproti účinnostiam pri dátových sadách nižšej hustoty, avšak opäť je zrejmá kladná závislosť

medzi rozmerom dátovej sady a efektívnosťou uvedených metód. Dodávame taktiež, že najvyššie časové úspory dosahujeme opäť použitím nami navrhnutého algoritmu.

Tabuľka 3, uvedená nižšie, poskytuje prehľad rýchlostí jednotlivých algoritmov, pre dátové sady rovnakých kardinalít a dimenzii ako v predošlých tabuľkách, avšak s rozdielnou hustotou každej dátovej sady, ktorá je na úrovni 25%, čo znamená, že došlo k takmer dvojnásobnému zvýšeniu počtu efektívnych útvarov oproti dátovým sadám použitých v Tabuľke 2. Oproti predchádzajúcim výsledkom môžeme pozorovať opätovný nárast celkového výpočtového času, čo je zapríčinené ďalším nárastom hustoty v dátovej sade.

dátový súbor	tradičný	RBE	HD-RBE	BH	nový
5000-5-25	137.14	89.51	89.43	79.81	83.93
5000-10-25	204.66	125.60	104.05	95.16	97.63
5000-15-25	263.98	166.11	121.13	112.59	116.37
5000-20-25	390.55	253.06	166.35	153.93	158.21
7500-5-25	346.22	199.20	151.64	146.60	154.61
7500-10-25	422.66	270.19	184.15	185.46	194.46
7500-15-25	757.23	391.15	238.99	234.99	248.25
7500-20-25	933.26	555.47	296.77	307.60	320.76
10000-5-25	630.89	393.57	243.38	247.73	263.82
10000-10-25	798.69	484.23	288.88	303.27	318.51
10000-15-25	1261.42	732.72	364.64	386.37	411.13
10000-20-25	1693.91	1034.33	498.35	533.02	566.24

Tabuľka 3: Informácie o výsledkoch prislúchajúcich dátovým sadám s hustotami 25% pre jednotlivé algoritmy

Z Tabuľky 9, uvedenej v prílohe, môžeme vidieť, že časové úspory algoritmu RBE oproti tradičnému sa nachádzajú v rozmedzí 34.73% - 48.34%. Časové úspory metódy hierarchickej dekompozície sa pohybujú v rozmedzí 34.79% - 71.09%, úspory použitím metódy Build Hull sú v rozmedzí 41.8% - 69.37% a napokon úspory nášho algoritmu sa nachádzajú v rozmedzí 38.8% - 67.41%, oproti tradičnému prístupu výpočtu.

Vyššie prezentované výsledky nám poskytli súhrnný prehlaď celkových výpočtových časov jednotlivých algoritmov na simulovaných dátach stredných kardinalít. Podotýkame,

že z prezentovaných výsledkov by každá dátová sada bola riešiteľná v rozumnom čase aj za použitia tradičného prístupu, avšak využitím predstavených algoritmov môžeme obdržať viditeľné časové úspory.

Pripomíname, že v tejto práci využívame aditívny model, ktorý patrí medzi najjednoduchších zástupcov obálkovej analýzy dát. Poznamenajme, že pri použití zložitejších modelov na výpočet efektivít, by celkový výpočtový čas mohol za použitia tradičného prístupu podstatne narásť. Predstavené algoritmy by mohli viesť v takýchto prípadoch k podstatne vyšším časovým úsporám oproti použitiu tradičného prístupu. Dodávame, že metóda hierarchickej dekompozície, metóda Buil Hull ako aj nami prezentovaný algoritmus, pozostávajú z 2 fáz a ako bolo uvedené v predchádzajúcej kapitole, táto vlastnosť predstavuje výhodu, keďže v spomínaných algoritmoch by zložitejší model bol aplikovaný až v 2. fáze, kde súčty v podmienkach prípustnosti budú prislúchať množine efektívnych útvarov.

V článku [13] je uvedené, že pre veľké dátové sety je typický nízky pomer efektívnych útvarov k celkovému počtu, teda nízka hustota. Rozhodli sme sa preto otestovať metódu hierarchickej dekompozície, metódu Build Hull a taktiež nami navrhnutú metódu na simulovaných dátach veľkých kardinalít 25000, 50000, 75000 a 100000 útvarov, pričom každá kardinalita je zastúpená dimenziami 5 (2 vstupy, 3 výstupy), 10 (5 vstupy, 5 výstupov), 15 (7 vstupov, 8 výstupov) a 20 (10 vstupov, 10 výstupov) a zároveň každá dátová sada má udávanú hustotu na úrovni 1%.

Z dôvodu časovej náročnosti sme pre nasledovné dátové sety nevykonávali výpočty pomocou tradičného prístupu ani pomocou algoritmu RBE. Pokúsili sme sa však odhadnúť výpočtový čas na jeden útvar použitím tradičného prístupu. Tento odhad sme vykonali spôsobom, že pre každú dátovu sadu sme vyrátali efektivity prvých 1000 útvarov a následne sme obdržané výsledky predelili počtom útvarov. Tým sme dostali odhady výpočtových časov na jeden útvar, ktoré prezentujeme v Tabuľke 4.

kardinalita	dimenzia=5	dimenzia=10	dimenzia=15	dimenzia=20
25000	0.19	0.36	0.43	0.46
50000	0.69	0.83	1.07	1.14
75000	1.16	1.42	1.9	2.24
100000	1.79	2.06	3.09	3.48

Tabuľka 4: Odhadnuté výpočtové časy na 1 útvar, pre dáta veľkých kardinalít

V Tabuľke 5 sú uvedené výsledky celkových časov algoritmov hierarchickej dekompozície, Build Hull ako aj nami navrhnutého algoritmu na dátach veľkých kardinalít. Z nižšie uvedenej tabuľky je zrejmé, že metóda hierarchickej dekompozície zaostáva za metódou Build Hull, ako aj nami prezentovaným algoritmom. Vyšší čas výpočtu pomocou použitia metódy hierarchickej dekompozície oproti použitiu algoritmu Build Hull ako aj nami navrhnutého algoritmu bolo badať už pri dátových sadách stredných kardinalít, s hustotami na úrovni 1%, prezentovaných v Tabuľke 1. Z predpokladu nízkej hustoty dochádzame k záveru, že použitie metódy Build Hull a nami navrhnutého algoritmu sa javí ako najefektívnejšie v dátových sadách veľkých rozmerov.

Tabuľka 10 uvedená v prílohe demonštrujeme časové úspory metódy hierarchickej dekompozície, Build Hull ako aj nášho algoritmu oproti odhadnutým výpočtovým časom za použitia tradičného prístupu, uvedených v Tabuľke 4. Z Tabuľky 10 môžeme vidieť, že časové úspory metódy hierarchickej dekompozície sa pohybujú v rozmedzí 90,78% - 98,47%, úspory použitím metódy Build Hull sa nachádzajú v rozmedzí 93,63% - 98,96% a napokon úspory nášho algoritmu sa nachádzajú v rozmedzí 93,92% - 98,98%.

Na stránke [9] sa nachádzajú taktiež reálne dáta kardinality 19939 a dimenzie 11 (6 vstupov, 5 výstupov), obsahujúce údaje o amerických komerčných bankách. Z tejto dátovej sady sme náhodným výberom vytvorili niekoľko ďalších redukovaných dátových vzoriek s rôznym počtom útvarov (17 500, 15 000, 12 500 a 10 000). Pre lepšiu demonštráciu efektívnosti algoritmov, sme z každého z týchto 5 dátových súborov (1 pôvodná a 4 vytvorené dodatočne náhodným výberom) vytvorili ešte dodatočné dátové vzorky nasledovným postupom. Najprv sme vzali prvé 4 vstupy a 4 výstupy, čím sme obdržali dimenziu 8, a teda ďalších 5 dátových súborov. Následne sme vybrali prvé 3 vstupy a 3 výstupy, čo nám pomohlo vytvoriť ešte 5 dátových sád s dimenziou 6. Týmto spôsobom

dátový súbor	HD-RBE	BH	nový
25000-5-01	427.01	294.99	281.42
25000-10-01	536.73	330.22	317.46
25000-15-01	710.49	392.10	433.68
25000-20-01	776.71	437.51	456.47
50000-5-01	928.95	709.21	690.98
50000-10-01	1213.52	879.73	833.23
50000-15-01	1707.21	997.71	1070.63
50000-20-01	1831.86	1122.42	1143.67
75000-5-01	1452.37	1208.06	1156.17
75000-10-01	1901.62	1519.68	1417.86
75000-15-01	3047.37	1836.07	1895.37
75000-20-01	3396.71	2221.90	2238.75
100000-5-01	2140.36	1849.66	1792.13
100000-10-01	2638.24	2203.49	2060.95
100000-15-01	4830.86	2950.64	3087.59
100000-20-01	5215.62	3561.21	3478.17

Tabuľka 5: Informácie o výsledkoch prislúchajúcich dátovým sadám s hustotami 1% pre jednotlivé algoritmy

sme obdržali celkovo 15 dátových sád rôznych rozmerov a hustôt.

Nasledovná Tabuľka 6 poskytuje prehľad rýchlostí jednotlivých algoritmov na reálnych dátach. Dodávame, že pôvodný dátový súbor, evalvovaný za pomoci tradičného prístupu, si vyžiadal približne 3 hodiny 17 minút, kým výpočty za pomoci algoritmu Build Hull, resp. nami prezentovaného algoritmu, si vyžiadali iba 14 resp. 17 minút, čo značí signifikantný pokles.

Z Tabuľky 11 uvedenej v prílohe môžeme vidieť, že časové úspory algoritmu RBE oproti tradičnému sa nachádzajú v rozmedzí 48.08% - 57.13%. Časové úspory metódy hierarchickej dekompozície sa pohybujú v rozmedzí 81.46% - 93.57%, úspory použitím metódy Build Hull v rozmedzí 85.39% - 94.81% a napokon úspory nášho algoritmu v rozmedzí 81.81% - 94.49%, oproti tradičnému prístupu výpočtu.

dátový súbor	tradičný	RBE	HD-RBE	BH	nový
19939-11-6.18	11822.52	5469.20	1076.19	820.75	992.46
19939-8-2.12	7771.83	3391.42	499.78	403.66	461.52
19939-6-0.99	5864.22	2514.00	408.42	307.02	323.25
17500-11-6.68	8726.41	4047.97	887.78	692.92	862.50
17500-8-2.42	5540.50	2507.41	433.98	346.59	392.53
17500-6-1.13	3863.68	1869.83	350.97	267.77	284.22
15000-11-7.30	6097.61	2903.03	702.93	571.90	701.37
15000-8-2.35	3891.17	1680.22	357.15	279.47	317.32
15000-6-1.20	2671.92	1315.07	302.48	229.66	236.42
12500-11-7.90	4053.27	1842.23	545.51	445.07	538.93
12500-8-2.72	2374.80	1174.47	295.74	234.24	262.75
12500-6-1.33	1812.91	941.24	247.64	185.71	195.66
10000-11-8.72	2240.06	1152.03	415.25	327.20	407.45
10000-8-2.75	1511.92	755.59	225.79	174.94	201.21
10000-6-1.57	1169.75	606.40	198.82	146.04	150.54

Tabuľka 6: Informácie o výsledkoch prislúchajúcich reálnym dátovým sadám pre jednotlivé algoritmy

3.2 Zhrnutie výsledkov

Na záver môžeme zhrnúť, že z uvedených výsledkov je pre každú dátovú sadu podrobenú analýzam zrejmý signifikantný pokles celkových výpočtových časov, za použitia predstavených algoritmov, čím sa potvrdzuje vhodnosť ich využitia ako na simulovaných tak aj na reálnych dátových sadách. Podotýkame, že pri dátach stredných veľkostí kardinalít, boli celkové výpočtové časy aj použitím tradičného prístupu celkom uspokojivé. Navyše ak uvážime, že by sme disponovali lepšími výpočtovými zdrojmi, pripadne evalvácie jednotlivých úloh by prebiehali v inom softvéry, mohlo by sa zdať využitie efektívnejších algoritmov ako neopodstatnené. Ak však uvážime fakt, že pre výpočty efektivít jednotlivých útvarov v dátových sadách sme v tejto práci využívali aditívny model, ktorý patrí medzi najjednoduchších zástupcov, zložitejší model by mohol viesť k podstatnému na-

výšeniu výpočtového času použitím tradičného prístupu. V takomto prípade by sa javilo použitie navrhovaných algoritmov za celkom opodstatnené. Ak by sme navyše mali obmedzený čas výpočtu považujeme použitie navrhovaných postupov za vhodné. Obmedzený čas evalvácie by mohol napríklad spôsobovať, jedna zo spomínaných aplikácii DEA modelov uvedená v článku [11], ktorý spomína aplikovateľnosť DEA modelov pri identifikáciách podvodov s platobnými kartami. Ak by sme v takom prípade mali obmedzený čas k evalváciiam navrhnuté postupy by sa mohli aj pri dátach stredných kardinalít javiť ako podstatne efektívnejšie. Pre dátové súbory veľkých kardinalít sa však javil tradičný prístup ako vysoko nepostačujúci. Pre takéto dátové sady pokladáme aplikáciu prezentovaných postupov za žiadúcu. Pričom k uvedenej aplikácie DEA modelov na podvody s platobnými kartami, bude dochádzať s najvyššou pravdepodobnosťou pri dátach veľkých kardinalít. Navyše pri spomínanej aplikácii sa operácie s platobnými kartami dynamicky menia, čiže aplikácia zlepšujúcich algoritmov je nutná.

Záver

Nebývalý technologický pokrok pozorovaný v súčasnej dobe so sebou prináša množstvá veľkých dátových setov, ktoré je potrebné analyzovať a vyhodnocovať. To, na čo nám kedysi postačovali zaužívané princípy, postupy a vtedajšia výpočtová sila, sa v budúcnosti môže ukázať ako nedostačujúce. Z tohto dôvodu sa natíska potreba vývoja nových postupov a riešení problémov, ktoré by viedli k efektívnejšiemu a najmä rýchlejšiemu vyhodnocovaniu väčších dátových setov v problematike DEA. V našej práci sme sa preto venovali popisu takýchto algoritmov pre oblasť DEA modelov.

V prvej kapitole práce sme sa venovali najmä objasneniu základných pojmov a zároveň všeobecnej terminológii ustálenej v DEA problematike, pričom sme zároveň predstavili jeden zo základných neorientovaných modelov, ktorým bol aditívny model. Následne sme predstavili jeho kľúčové vlastnosti a zároveň sme spomenuli taktiež jeho základné výhody podstatné pre problematiku tejto práce. Súčasťou prvej časti práce bolo taktiež predstavenie istej modifikácie aditívneho modelu vedúcej k výpočtu tzv. superefektivity, ktorej sme sa venovali v časti 1.5.3. Prínosom tejto časti práce bola najmä formulácia niektorých tvrdení s ich príslušnými dôkazmi, ktoré nám neskôr poslúžili k vytvoreniu vlastného algoritmu slúžiacemu k redukcii výpočtového času.

V druhej kapitole sme sa primárne zamerali najmä na dôkladné spracovanie literatúry, týkajúcej sa riešení DEA modelov v prípade veľkých dátových súborov, pričom pri spracovaní tejto kapitoly sme čerpali najmä z publikácií [1], [3], [8]. Predstavili sme tri najčastejšie skloňované algoritmy, menovite metódu RBE, metódu Hierarchickej dekompozície a metódu Build Hull. Ďalším prínosom tejto kapitoly bolo navrhnutie vlastného prístupu k riešeniu DEA modelov v prípade veľkých dátových setov prezentovaný v časti ref, ktorý využíval k zrýchleniu výpočtu modifikovaný aditívny model pre výpočet superefektivity, a zároveň isté princípy, ktoré sme nadobudli oboznámením sa s predchádzajúcimi predstavenými algoritmami.

Posledná kapitola tejto práce bola založená na empirickom porovnaní predstavených algoritmov na veľkých dátových sadách rôznych kardinalít, dimenzií a hustôt. Súčasťou tejto kapitoly bola aj následná analýza jednotlivých algoritmov s vyhodnoteniami, ktoré sa javili ako uspokojivé, vzhľadom k súčasným potrebám DEA modelov. Na konci tejto časti sme taktiež uviedli možnosti využitia prezentovaných algoritmov.

Dnešné časy sa vyznačujú rýchlymi zmenami, pri ktorých sa nedá s istotou povedať, čo v budúcnosti nastane. Z toho dôvodu sa ani nami prezentované algoritmy nemusia v budúcnosti javiť ako dostačujúce. Preto ako možné rozšírenie tejto práce vidíme práve možnú kombináciu algoritmov prezentovaných v tejto práci za účelom ďalšieho zefektívnenia výpočtov.

Zoznam použitej literatúry

[1] Ali, A.I.: *Streamlined computation for data envelopment analysis*, European Journal of Operational Research 64 (1993), 61-67

[2] Banker, R.D., Charnes, A., Cooper, W.W.: *Some Models for Estimating Technical and Scale Inefficiencies in Data Envelopment Analysis*, Management Science 30 (1984), 1078-1092

[3] Barr, R.S., Durchholz, M.L.: *Parallel and Hierarchical Decomposition Approaches for Solving Large-Scale Data Envelopment Analysis Models*, Annals of Operations Research 73 (1997), 339–372

[4] Charnes, A., Cooper, W.W., Golany, B., Seiford, L., Stutz, J.: *Foundations of data envelopment analysis for Pareto-Koopmans efficient empirical production functions*, Journal of Econometrics 30 (1985), 91-107

[5] Charnes, A., Cooper, W.W., Rhodes, E.: *Measuring the efficiency of decision making units*, European Journal of Operational Research 2 (1978), 429-444

[6] Du, J., Liang, L., Zhu, J.: *A slacks-based measure of super-efficiency in data envelopment analysis: A comment*, European Journal of Operational Research 204 (2010), 694–697

[7] Dulá, J.H.: *A computational study of DEA with massive data sets*, Computers Operations Research 35 (2008), 1191–1203

[8] Dulá, J.H.: *An Algorithm for Data Envelopement Analysis*, INFORMS Journal on Computing 23 (2011), 284-296

[9] Dulá, J.H.: *José Dulá's Home page*, dostupné na internete: http://www.people.vcu.edu/ jdula/

[10] Dulá, J.H., López, F.J.: *Algorithms for the Frame of a Finitely Generated Unbounded Polyhedron*, INFORMS Journal on Computing 18 (2006), 97–110

[11] Dulá, J.H., López, F.J.: *DEA with streaming data*, Omega 41 (2013), 41-47

[12] Dulá, J.H., López, F.J.: *Preprocessing DEA*, Computers Operations Research 36 (2009), 1204–1220

[13] Dulá, J.H., Thrall, R.M.: *A Computational Framework for Accelerating DEA*, Journal of Productivity Analysis 16 (2001), 63–78

[14] Halická, M.: *DEA modely*, učebné texty, FMFI UK, Bratislava, 2015, dostupné na internete (14.11.2020):
http://www.iam.fmph.uniba.sk/institute/halicka/teach/DEAskripta.pdf

[15] Halická, M., Trnovská, M.: *A unified approach to non-radial graph models in data envelopment analysis: common features, geometry, and duality*, European Journal of Operational Research 289 (2020), 611-627

[16] Halická, M., Trnovská, M.: *Negative features of hyperbolic and directional distance models for technologies with undesirable outputs*, Central European Journal of Operations Research 26 (2018), 887-907

[17] Halická, M., Trnovská, M.: *The Russell measure model: Computational aspects, duality, and profit efficiency*, European Journal of Operational Research 268 (2018), 386-397

[18] Khezrimotlagh, D., Zhu, J., Cook, W.D., Toloo, M.: *Data Envelopment Analysis and Big Data*, European Journal of Operational Research 274 (2019), 1047-1054

[19] Plesník, J., Dupačová, J., Vlach, M.: *Lineárne programovanie*, Alfa, Bratislava, 1998

[20] Siems, T.F., Barr, R.S.: *Benchmarking the Productive Efficiency of U.S. Banks*, Financial Industry Studies, Federal Reserve Bank of Dallas (1998), 11-24

[21] Sueyoshi, T.: *A Special Algorithm for an Additive Model in Data Envelopment Analysis*, Journal of the Operational Research Society 41 (1990), 249-251

[22] Sueyoshi, T., Chang, Y.: *Efficient algorithm for additive and multiplicative models in data envelopment analysis*, Operations Research Letters 8 (1989), 205-213

[23] Tone, K.: *A slacks-based measure of efficiency in data envelopment analysis*, European Journal of Operational Research 130 (2001), 498-509

Príloha A

dátový súbor	RBE	HD-RBE	BH	nový
5000-5-01	39.41	46.91	60.35	62.45
5000-10-01	39.82	54.69	65.46	66.85
5000-15-01	43.32	63.02	75.22	75.99
5000-20-01	52.05	75.07	85.11	85.68
7500-5-01	46.03	66.02	72.12	73.45
7500-10-01	42.03	70.23	77.12	77.90
7500-15-01	49.69	77.59	84.62	85.28
7500-20-01	55.80	87.08	91.71	92.12
10000-5-01	45.92	74.07	78.92	80.15
10000-10-01	45.43	76.88	82.00	82.74
10000-15-01	58.77	86.70	90.36	90.83
10000-20-01	53.25	89.28	92.87	93.37

Tabuľka 7: Informácie o relatívnych časových úsporách jednotlivých algoritmov, oproti použitiu tradičného prístupu pre datové sady s hustotami 1%.

dátový súbor	RBE	HD-RBE	BH	nový
5000-5-13	38.43	46.37	55.49	55.93
5000-10-13	40.77	55.59	62.84	63.81
5000-15-13	40,28	62.40	67.15	67.83
5000-20-13	36.38	60.27	66.18	66.63
7500-5-13	44.57	62.30	65.55	66.06
7500-10-13	39.70	64.92	67.32	68.58
7500-15-13	48.11	74.00	75.65	76.16
7500-20-13	45.28	75.28	76.36	76.81
10000-5-13	42.52	66.52	68.36	68.61
10000-10-13	40.42	71.27	72.14	72.82
10000-15-13	45.57	79.09	79.13	79.51
10000-20-13	43.76	81.39	81.18	81.57

Tabuľka 8: Informácie o relatívnych časových úsporách jednotlivých algoritmov, oproti použitiu tradičného prístupu pre datové sady s hustotami 13%.

dátový súbor	RBE	HD-RBE	BH	nový
5000-5-25	34.73	34.79	41.80	38.80
5000-10-25	38.63	49.16	53.50	52.30
5000-15-25	37.07	54.11	57.35	55.92
5000-20-25	35.20	57.41	60.59	59.49
7500-5-25	42.46	56.20	57.66	55.34
7500-10-25	36.07	56.43	56.12	53.99
7500-15-25	48.34	68.44	68.97	67.22
7500-20-25	40.48	68.20	67.04	65.63
10000-5-25	37.62	61.42	60.73	58.18
10000-10-25	39.37	63.83	62.03	60.12
10000-15-25	41.91	71.09	69.37	67.41
10000-20-25	38.94	70.58	68.53	66.57

Tabuľka 9: Informácie o relatívnych časových úsporách jednotlivých algoritmov, oproti použitiu tradičného prístupu pre datové sady s hustotami 25%.

dátový súbor	HD-RBE	BH	nový
25000-5-01	90.78	93.63	93.92
25000-10-01	93.97	96.29	96.44
25000-15-01	94.86	97.16	96.86
25000-20-01	96.37	97.96	97.87
50000-5-01	96.00	96.95	97.03
50000-10-01	97.05	97.86	97.97
50000-15-01	97.44	98.51	98.40
50000-20-01	97.80	98.65	98.63
75000-5-01	97.31	97.77	97.86
75000-10-01	98.06	98.45	98.55
75000-15-01	97.91	98.74	98.70
75000-20-01	97.75	98.53	98.52
100000-5-01	97.87	98.16	98.22
100000-10-01	98.43	98.69	98.78
100000-15-01	98.24	98.93	98.88
100000-20-01	98.47	98.96	98.98

Tabuľka 10: Informácie o relatívnych časových úsporách pre jednotlivé algoritmy, oproti odhadnutým výpočtovým časom za použitia tradičného prístupu, pre datové sady veľkých kardinalít s hustotami 1%

dátový súbor	RBE	HD-RBE	BH	nový
19939-11-6.18	53.74	90.90	93.06	91.61
19939-8-2.12	56.36	93.57	94.81	94.06
19939-6-0.99	57.13	93.04	94.76	94.49
17500-11-6.68	53.61	89.83	92.06	90.12
17500-8-2.42	54.74	92.17	93.74	92.92
17500-6-1.13	51.60	90.92	93.07	92.64
15000-11-7.30	52.39	88.47	90.62	88.50
15000-8-3.35	56.82	90.82	92.82	91.85
15000-6-1.20	50.78	88.68	91.40	91.15
12500-11-7.90	54.55	86.54	89.02	86.70
12500-8-2.72	50.54	87.55	90.14	88.94
12500-6-1.33	48.08	86.34	89.76	89.21
10000-11-8.72	48.57	81.46	85.39	81.81
10000-8-2.75	50.02	85.07	88.43	86.69
10000-6-1.57	48.16	83.00	87.52	87.13

Tabuľka 11: Informácie o relatívnych časových úsporách jednotlivých algoritmov, oproti použitiu tradičného prístupu pre reálne datové sady.