

VYSOKÁ ŠKOLA EKONOMICKÁ V PRAZE

FAKULTA INFORMATIKY A STATISTIKY

KATEDRA EKONOMETRIE

**tutORial – on-line systém pro výuku
operačního výzkumu**

Lucie Svobodová

Vedoucí bakalářské práce: Prof. Ing. Josef Jablonský, CSc.

PRAHA 2007

Prohlášení

Prohlašuji, že jsem bakalářskou práci na
téma „tutORial – on-line systém pro výuku
operačního výzkumu“ vypracovala samostatně.
Použitou literaturu a podkladové materiály
uvádím v přiloženém seznamu literatury.

v Praze dne 20. 12. 2006

.....

Lucie Svobodová

Obsah:

OBSAH:	3
1 ÚVOD	5
2 MODEL Y PROJEKTU TUTORIAL	6
2.1 OBE CNÁ CHARAKTERISTIKA	6
2.1.1 Model y dynamického programování	7
2.1.2 Optimalizace v grafech	9
2.1.3 Celočíselné programování	11
2.1.4 Lineární algebra	11
2.1.5 Lineární programování	13
2.1.6 Simulace	13
2.2 ZHODNOCENÍ MODULŮ	14
3 ŘEŠENÍ ÚLOH LINEÁRNÍHO PROGRAMOVÁNÍ V RÁMCI PROJEKTU TUTORIAL	15
3.1 OBE CNÉ PŘEDPOKLADY A VÝCHODISKA	15
3.2 SIMPLEXOVÁ METODA	15
3.2.1 Gaussova eliminační metoda	16
3.2.2 Podmínky nezápornosti	17
3.2.3 Účelová funkce	18
3.2.4 Optimalizace	18
3.2.5 Speciální případy	20
3.2.6 Dvoufázová simplexová metoda	21
3.2.7 Simplexové řešitele	21
3.3 VIRTUÁLNÍ DUALITA	22
3.4 DUÁLNÍ SIMPLEXOVÁ METODA A DOPRAVNÍ PROBLÉM	22
3.5 MAĎARSKÁ METODA	22
3.6 ZHODNOCENÍ MODULŮ LINEÁRNÍHO PROGRAMOVÁNÍ	22
4 CELOČÍSELNÉ LINEÁRNÍ PROGRAMOVÁNÍ V RÁMCI PROJEKTU TUTORIAL	24
4.1 OBE CNÉ PŘEDPOKLADY A VÝCHODISKA	24
4.2 ÚLOHA O BATOHU (THE KNAPSACK PROBLEM)	24
4.2.1 Neomezený problém	24
4.2.2 Neomezený problém – dynamické programování	26
4.3 GOMORYHO METODA ŘEZNÝCH NADROVIN	27
4.4 ÚLOHA OSMI DAM	27
4.5 ÚLOHA OSMI DAM - NOVĚ	28
4.6 8 SNADNÝCH KUSŮ (8 EASY PIECES)	29
4.7 ZHODNOCENÍ MODULŮ CELOČÍSELNÉHO PROGRAMOVÁNÍ	30

<u>5 ZÁVĚR.....</u>	<u>31</u>
POUŽITÁ LITERATURA.....	32
KNIHY A UČEBNICE.....	32
INTERNETOVÉ STRÁNKY.....	32
SEZNAM OBRÁZKŮ.....	33
SEZNAM PŘÍLOH.....	34

1 Úvod

Cílem mé práce je představit projekt TutORial, který se na internetu snaží přiblížit modely operačního výzkumu širší veřejnosti.

Tento projekt je zvláštní iniciativou IFORS (*International Federation of Operational Research Societies*), jejímž záměrem je podnítit mezinárodní spolupráci v oblasti operačního výzkumu. Jedná se o mezinárodní společnost pro operační výzkum, které zastřešuje národní společnosti operačního výzkumu z více než 50 zemí, a to z geografického uskupení Pacifická Asie, Evropa, Severní Amerika a Jižní Amerika. IFORS si klade za cíl podpořit OR jako akademickou disciplínu a profesi.

Cílem projektu TutORial, je během tří následujících let, rozvinout komplexní web založený na on-line výukovém systému pro operační výzkum.

Tento systém nabízí široké spektrum možností. Příkladem může být:

- Rozvinutí nových interaktivních modelů pro metody a techniky z OR (*operational research*)/ MS (*management science*)
- Zvýšení možností již existujících modelů
- Obsahové zlepšení materiálů na podporu stávajících modelů
- Rozvinutí on-line pomocného příslušenství pro existující modely
- Testování modelů na chyby, soudržnost a optimálnost pro uživatele

Práci rozdělím celkem do pěti hlavních oddílů, z nichž první je úvod. Ve druhém oddíle stručně představím modely projektu TutORial. Ve třetím a ve čtvrtém oddíle se budu podrobněji věnovat konkrétním postupům při řešení úloh lineárního programování a celočíselného lineárního programování.

Výsledky celé práce shrnu a zhodnotím v závěru.

2 Modely projektu TutORial

2.1 Obecná charakteristika

Projekt tutORial si klade za cíl rozvinout rozsáhlou knihovnu samostatných entit/modelů. Každý z těchto modelů je určen konkrétnímu tématu v rámci OR/MS. Některé z modelů jsou natolik propracované a detailní, že se pro větší přehlednost skládají z dalších sub-modelů.

I když se mnohé z nich budou výrazně odlišovat velikostí, tvarem, obsahem, strukturou, stylem, atd., budou mít následující typické znaky:

- Text
popis problému, metoda a techniky, atd.
- Podoba
obrázky, fotky, kresby, kliparty
- Výpočetní mechanismus
samostatný počítačový kód vykonávající výpočty a zadané úlohy
- Animace
dynamický textový nebo obrázkový displej
- Uživatelské rozhraní
pozadí, nabídka, tlačítka, atd., která tak umožní uživateli ovlivňovat předměty modelu (např. vkládat data,...)

TutORial se zaměřuje na modely, které obsahují nejenom text, ale i výpočetní a animační nástroje.

Rozdělení modelů do skupin podle oboru zaměření

Modely jsou stejně jako v realitě každý zaměřený na jednu specifickou problematiku. Aby byl přístup tutORialu co nejpřehlednější, autoři modely rozdělili dle oboru působnosti.

Struktura členění odpovídá jednotlivým okruhům OR/MS disciplín: (členění včetně konkrétních názvů modelů)

1. Dynamické programování

Modely: Die Hard at the Pub, Towers of the Hanoi, New Towers of Hanoi, Nejkratší cesta, Dijkstra's algorithm, a další..

2. *Optimalizace v grafech*

Modely: Nejkratší cesta, Dijkstra's algorithm, Topologické třídění, Přechodný závěr

3. *Celočíselné programování*

Modely: Větvení, Gomoryho metoda řezných nadrovin, Problém osmi dam, Problém osmi dam - nově, 8 Easy pieces

4. *Lineární algebra*

Modely: Row operations, Lineární rovnice, Inverze, Determinanty

5. *Lineární programování*

Modely: Simplexova metoda, Virtuální dualita, Duální simplexová metoda, Dopravní problém, Maďarská metoda

6. *Simulace*

Modely: M/M/1 Queues, Model G/G/s, Queueing Networks, Testování náhodných čísel

7. *Ostatní*

2.1.1 Modely dynamického programování

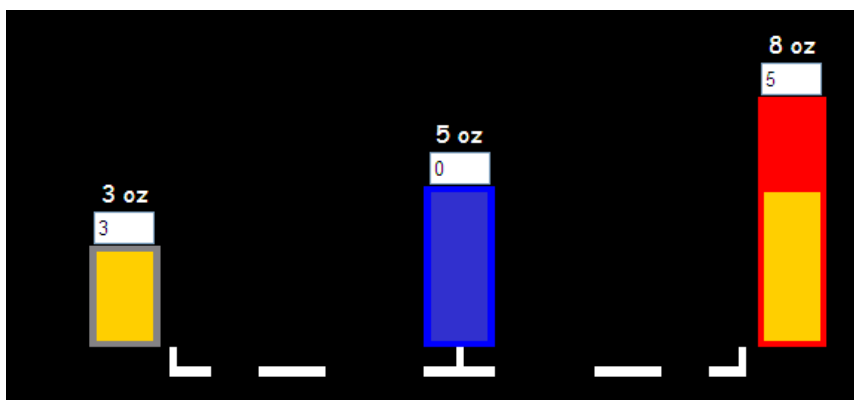
Dynamické programování spočívá v rozložení těžko řešitelného matematického modelu na několik dílčích problémů, které lze vyřešit samostatně, a tím pádem snadněji. Vyřešení dílčího problému zároveň nabízí optimální řešení i pro problém výchozí.¹

Pro konkrétnější představu uvedu dva příklady modelů, které jsou nabídnuty veřejně v projektu tutORial.

2.1.1.1 Die hard at the pub

Název tohoto modelu bychom mohli volně přeložit jako „*těžké umírání v hospodě*“. Jedná se o velmi známou praktickou hádanku, která existuje v několika verzích. Nejznámější z nich je verze, která se objevila ve filmu „Die hard with a vengeance“ (Smrtonosná past 2).

¹ Anderson, Sweeney, Williams: Introduction to management science, West publishing, 1994



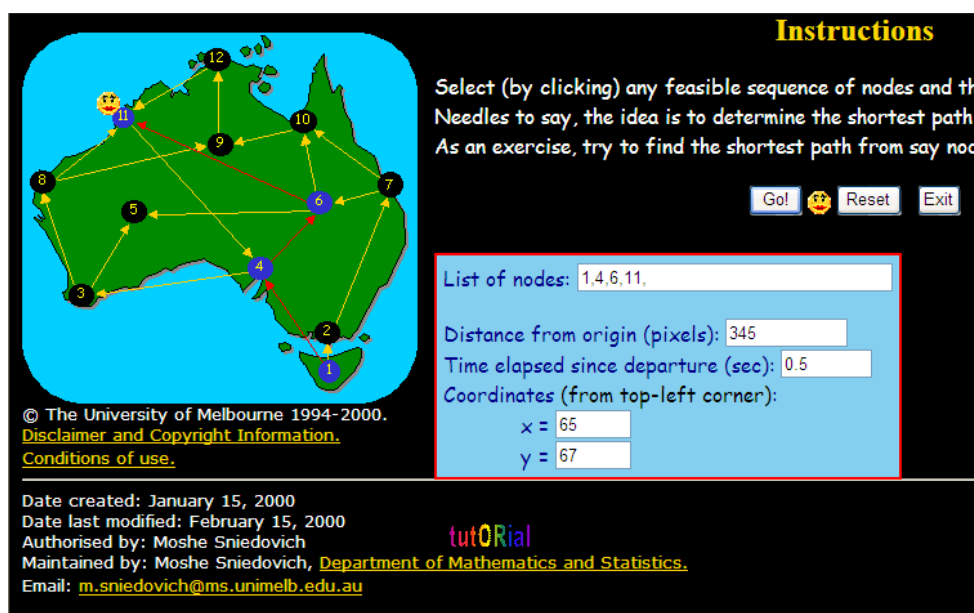
Obrázek 2-1 - náhled praktické aplikace "Die Hard at the Pub"

Hra spočívá v nalezení optimálního rozložení osmi jednotek tekutiny do dvou ze tří válců. Válce obsahují osm, pět a tři jednotky. Začínáme s osmi jednotkami tekutiny v největším válci a přeléváme můžeme vždy buď celý obsah válce, nebo do zaplnění válce.

Program tutORial nabízí možnost si hru vyzkoušet pomocí interaktivní internetové aplikace, ale hlavně nabízí detailní teoretickou analýzu problému a návod, jak lze situaci vyřešit pomocí vhodného modelu dynamického programování.

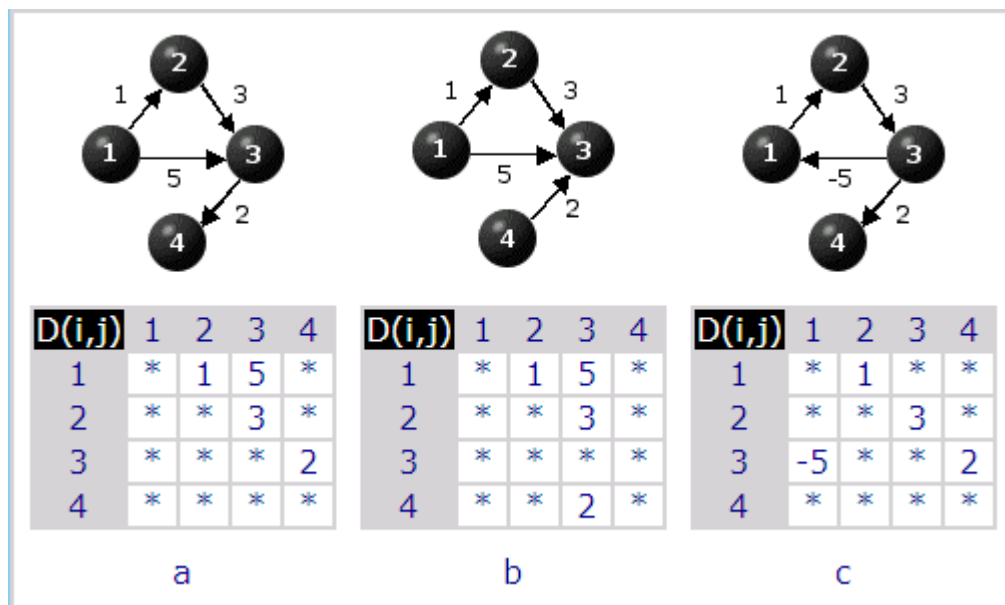
2.1.1.2 Nejkratší cesta v grafu

Jeden z nejznámějších modelů vůbec. Jedná se o převážně o logistický problém a to, jak se dopravit co nejkratší cestou z místa A do místa B, přes určité klíčové body. Tyto body se nazývají uzly a my máme rozhodnout, přes které uzly pojedeme tak, aby byla cesta co nejkratší.



Obrázek 2-2 - praktická ukázka problému "nejkratší cesta"

Projekt tutORial nám opět nabízí praktickou aplikaci, ve které si můžeme problém vyřešit, aniž bychom znali základy dynamického programování. Stejně jako model předchozí ale nabízí i detailní analýzu problému spolu s názorným popisem řešení s využitím dynamického programování.



Obrázek 2-3 - diagramy a matice modelu "nejkratší cesta"

2.1.2 Optimalizace v grafech

Tato sekce projektu tutORial jen odkazuje na možné řešení problémů grafickou cestou.

Dva modely jsou popsány i v rámci ostatních sekcí, proto bych ráda uvedla druhé dva modely.

2.1.2.1 Topologické třídění

Model navazující na již dříve uváděný problém: Nejkratší cesta. Tento model, jak již z názvu vyplývá, bude používat tzv. topologické třídění.

Pomáhá v usnadnění práce s modely nejkratší cesty. Abychom mohli dobře pracovat s uzly je důležité je seřadit podle důležitosti.(topologicky setřídít)

Projekt tutORial opět názorně naznačuje a detailně popisuje, jak v takovémto případě postupovat a dává tak konkrétní návod pro konkrétní využití.

We now apply this algorithm to the problem represented by the following graph. We shall replace the given labels A,B, ..., G by 1,2,...,7.

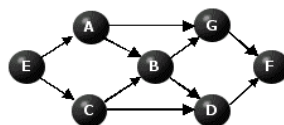


Figure 1

There is one node that has not been re-labeled yet having no immediate predecessors, namely E. We re-label it, 1, and deactivate the arcs emanating from it.

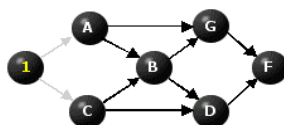


Figure 2

Obrázek 2-4 - náhled na tutORial návod pro topologické třídění

2.1.2.2 Tranzitivní závěr

Tranzitivní lze jinak říci jako přechodný.

Přechodnost je jev, který úzce souvisí s rozhodovacím procesem. v modelu je velmi dobře vysvětleno, co termín tranzitivní – přechodný znamená. Např. Pokud víme, že Marie je Pavlův následník (dědic) a Petr je Mariin dědic, pak můžeme říct, že Petr je zároveň i dědic Pavlův. Takovýchto jevů bychom našli určitě i více. Já pro příklad uvádím i netranzitivní proces: Honza je Pavlův syn a Petr je Honzův syn. To ale vůbec neznamená, že by i Petr byl Pavlův syn.(netranzitivní proces)

v modelu je vysvětlen celý postup názorného příkladu.

2.1.3 Celočíselné programování

Této sekci se budu velmi podrobně věnovat v další části mé práce, kde velmi podrobně popíši fungování modelů, spadajících právě do této sekce. Nyní bych se ráda zmínila jen o všeobecných charakteristikách celočíselného programování.

Jak už z názvu vyplývá, budeme pracovat s úlohami, které obsahují celá čísla, tzn. Úloha LP obsahuje podmínky celočíselnosti.

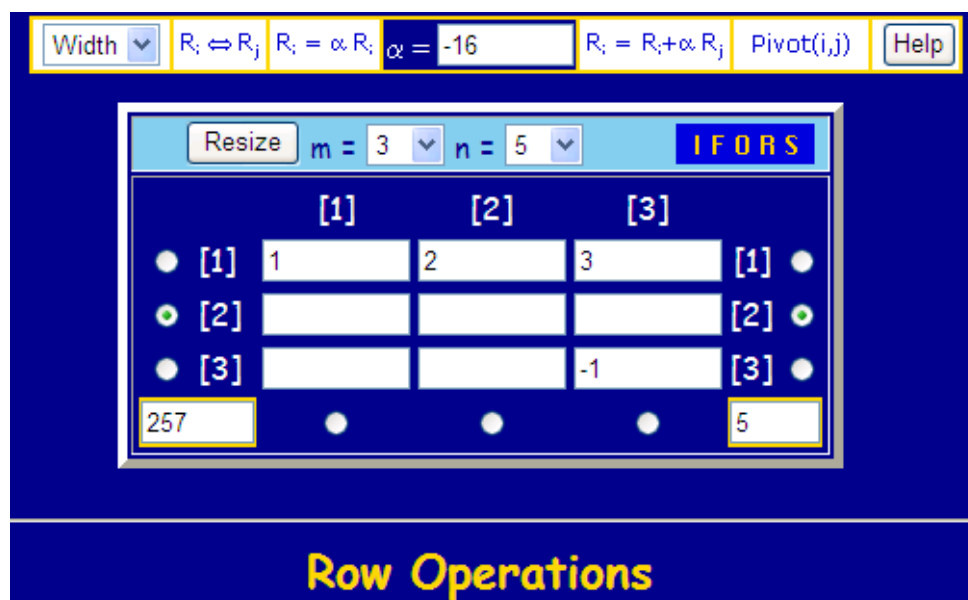
Tyto podmínky vyplývají většinou bezprostředně z formulace ekonomického modelu daného problému.²

2.1.4 Lineární algebra

2.1.4.1 Row operations (řádkové operace matic)

v modelu tutORial jen velmi uživatelsky jednoduchá aplikace, která nabízí možnost řešení matic.

Nelze nechat vyřešit matici jako takovou, ale aplikace umožňuje velmi snadno provádět řádkové úpravy matic (násobení, sčítání, odčítání, dělení, vytýkání...) Pro studenta, který se připravuje na bakalářskou zkoušku z matematiky a nechce se mu matice řádkově upravovat manuálně, by toto byla velmi užitečná pomůcka.



Obrázek 2-5 - hl. okno programu, který dokáže provádět řádkové úpravy matic

² J. JABLONSKÝ: Operační výzkum, Professional publishing, 2002, str.113

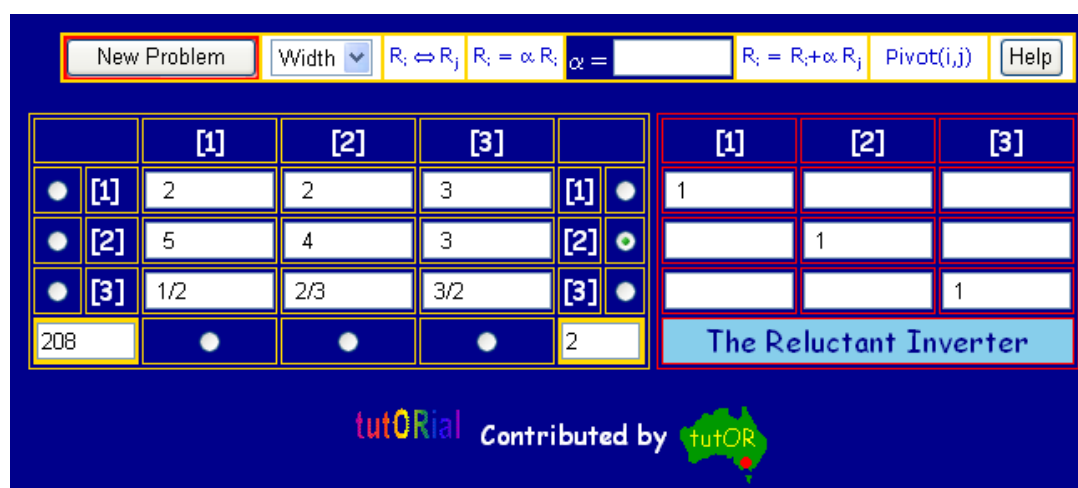
2.1.4.2 Řešení lineárních rovnic (The equator module)

Je interaktivní webová aplikace pro řešení soustav lineárních rovnic.

Funguje na podobném principu jako modul pro řádkové operace, tj. řeší soustavu lineárních rovnic, tento ale provádí i úpravy pravých stran. Stejně jako modul předchozí nedokáže vyřešit soustavu rovnic kompletně, ale velmi snadno provádí řádkové i sloupcové úpravy.

2.1.4.3 Inverze matic, determinanty

Další aplikace umožňující snadné řešení maticových problémů. „Inverter“, dokáže upravovat matici tak, abychom vypočítali, matici inverzní. „Determinator“, pomůže upravit matici tak, abychom snadno spočítali determinant.



Obrázek 2-6 - hl. okno programu, který vytváří inverzní matici



Obrázek 2-7 - hl. okno programu, který počítá determinant

2.1.5 Lineární programování

Tomuto problému bude věnována samostatná kapitola, proto bych se nyní zaměřila jen na všeobecnou definici.

Lineární programování je disciplína operačního výzkumu, která se zabývá řešením rozhodovacích problémů, ve kterých jde o určení intenzit realizace procesů, které probíhají nebo mohou probíhat v daném systému. Je prostředkem pro plánování realizace určitých procesů, který zabezpečuje dosažení optimálního výsledku ve vztahu k definovanému cíli.³

2.1.6 Simulace

Tato část se zabývá popisem a charakteristikou simulací pro modely front.

2.1.6.1 M/M/1 Queues (Jednoduchý exponenciální model hromadné obsluhy)

Aplikace v programu tutORial nabízí dvě možnosti řešení: řešení problému pomocí matematického modelu nebo pomocí simulace.

Opět máme na webu k dispozici jednoduchou aplikaci, která nám snadno spočítá daný problém. Stačí zadat intenzitu příchodů, intenzitu obsluhy, maximální délku fronty a délku pokusu.

2.1.6.2 Model G/G/s

Blok obsahuje dva simulátory pro jednoduchou frontu.

První z nich umožňuje simulaci systému jedné fronty s jedním příchozem a jednou službou. Uživatel si pro interval příchodu a trvání služby zvolí teoretické rozdělení (distribuci) nebo vloží rozdělení v tabulkové podobě. (tj. buď pravděpodobnosti nebo kumulativní rozdělení)

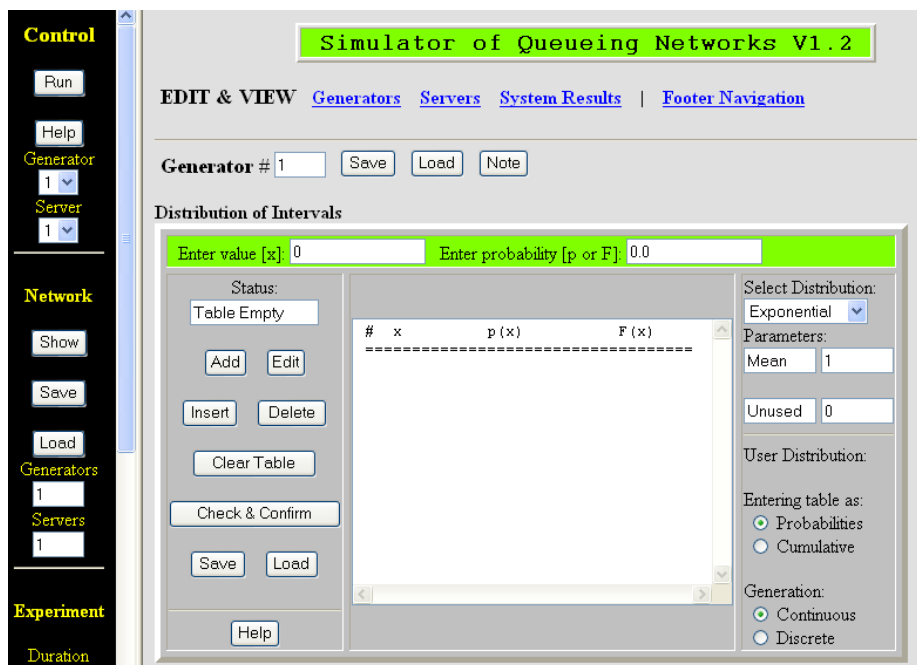
Druhý simulátor slouží k popisu a výpočtům systému jedné fronty s příchody v dávkách a dávkou služeb. Velikost dávky může být fixní nebo náhodná. Obslužné(á) místo(a) (služba) může buďto čekat na kompletní dávku nebo alternativně na dodávku částečnou.

2.1.6.3 Generování a testování náhodných čísel

Jsou posledními dvěma oddíly v rámci simulací.

³ J. JABLONSKÝ: Operační výzkum, Professional publishing, 2002, str.19

Queueing networks, jak již anglický název napovídá, je systém front uspořádaných do několika sítí. Jedná se o komplikovanou aplikaci, kterou lze využít pro řešení obtížných úloh hromadné obsluhy. Pro zajímavost uvádím ilustrativní obrázek pouze jedné z částí simulačního modulu.



Obrázek 2-8 - náhled na aplikaci pomáhající při výpočtech problémů hromadné obsluhy

Random Generator Tester je jednoduchá výpočetní aplikace umožňující testovat náhodný výběr. Výsledkem je jednorozměrná tabulka rozdělení četností vytvořená s odpovídající chí-square hodnotou.

2.2 Zhodnocení modulů

Všechny z výše uvedených modulů (s výjimkou modulů Lineárního programování a celočíselného programování, kterým se budu věnovat později) jsou velmi dobře teoreticky rozpracovány a i jejich praktická využitelnost je na velmi dobré úrovni.

Některé z modulů, jako např. Inverter a Derminator bych doporučila k praktickému využití nejen pro OR/MS disciplíny, ale i zejména matematickým disciplínám, kde se determinanty a inverzní matice počítají velmi často.

3 Řešení úloh lineárního programování v rámci projektu tutORial

3.1 Obecné předpoklady a východiska⁴

Abych se mohla věnovat praktickému popisu modulů projektu tutORial, je potřeba se zmínit o postupech používaných standardně při řešení úloh LP.

v praxi se začíná vždy formulací ekonomického modelu úlohy lineárního programování. Ten zpravidla obsahuje:

- cíl analýzy
- popis procesů, které v daném systému probíhají a mají vliv na definovaný cíl
- vliv činitelů, ovlivňujících procesy
- popsané vztahy mezi procesy, činiteli a cílem

Pro výpočet úlohy a nalezení optimálního řešení poté potřebujeme z modelu ekonomického stanovit ještě model matematický. Protože matematický model vychází z modelu ekonomického, má i obdobnou strukturu:

- cíl analýzy je vyjádřen účelovou funkcí
- každému procesu je přiřazena proměnná
- činitelům odpovídají lineární rovnice a nerovnice

3.2 Simplexová metoda

Simplexová metoda je kreační výpočetní postup pro nalezení optimálního řešení úlohy lineárního programování.

Postup výpočtu pomocí simplexové metody lze rozdělit na dvě základní fáze:

- I. fáze: nalezení výchozího základního řešení,
- II. fáze: iterační postup vedoucí k optimalizaci účelové funkce z.⁵

Simplexová metoda v pojetí projektu tutORial je popsána v oddíle: The Simplex Place. Jelikož jde o velmi rozsáhlou problematiku, modul je rozdělen do několika sub-modulů.

⁴ J. JABLONSKÝ: Operační výzkum, Professional publishing, 2002

⁵ J. JABLONSKÝ: Operační výzkum, Professional publishing, 2002, str.50

3.2.1 Gaussova eliminační metoda

Prvním sub-modulem je modul zabývající se Gaussovou eliminační metodou.

Postup v projektu tutORial velmi dobře a jednoduše popisuje základní pravidla při výpočtu soustavy rovnic nutných pro výpočet výchozí simplexové tabulky.

$$\begin{array}{rcl} 2x_1 + x_2 + x_3 & = & 40 \\ x_1 + x_2 + x_4 & = & 30 \\ x_1 + x_5 & = & 15 \end{array}$$

x_1	x_2	x_3	x_4	x_5	RHS
2	1	1	0	0	40
1	1	0	1	0	30
1	0	0	0	1	15

Obrázek 3-9 - výchozí soustava rovnic a její zobrazení v projektu tutORial

Tzn. již na začátku zvolená soustava rovnic je v kanonickém tvaru, tj. v tomto případě proměnným x_3 , x_4 , a x_5 odpovídá jednotkový vektor, který poskytuje výsledek $x_3=40$, $x_4=30$, $x_5=15$, nebo jinak vektor $x = (0,0,40,30,15)$.

Ne vždy, ale v praxi dochází k nalezení základního řešení tak snadno. Pokud soustava rovnic neobsahuje danou jednotkovou matici, je potřeba ji upravit následujícím způsobem. Určíme klíčový sloupec – vstupující proměnná a klíčový řádek – vystupující proměnná. Průsečíkem těchto dvou je pak klíčový prvek. Abychom pak nemuseli provádět dlouhý výpočet, použijeme interaktivní výpočetní mechanismus projektu tutORial.

x_1	x_2	x_3	x_4	x_5	RHS	
2	1	1	0	0	40	☐
1	1	0	1	0	30	☒
1	0	0	0	1	15	☐
☐	☒	☐	☐	☐	Pivot	Reset

Obrázek 3-10 - výchozí zadání do výpočetního mechanismu Gaussova eliminačního procesu; určení klíčového sloupce a klíčového řádku

Do výpočetního mechanismu jsme vložili původní matici, označili klíčový sloupec (x_2), klíčový řádek (2) a stiskli klávesu „Pivot“. Dostáváme pak nové základní řešení s vlevo naznačenými základními proměnnými, x_2 , x_3 a x_5 , čemuž odpovídá řešení $x = (0,30,10,0,15)$.

BV	x_1	x_2	x_3	x_4	x_5	RHS	
☒ 3	1		1	-1		10	☐
☒ 2	1	1		1		30	☒
☒ 5	1	0	0	0	1	15	☐
	☐	☒	☐	☐	☐	Pivot Reset	

Obrázek 3-11 - výsledek ve výpočetním mechanismu Gaussova eliminačního procesu; BV – označeny základní proměnné; RHS – pravá strana

3.2.2 Podmínky nezápornosti

Jedním z hlavních omezení a předpokladů simplexové metody je nezápornost proměnných. (non-negativity)

Toho lze dosáhnout několika způsoby:

- proměnné v prvotním základním řešení jsou nezáporné
- proměnné v jakémkoliv další řešení vytvořeném při úpravách zůstávají nezáporné

Jak tohoto předpokladu dosáhneme ale v praxi? Zejména druhé z nich, kdy nevíme, jakou proměnou (sloupec) je potřeba nahradit, tak aby sloupec pravých stran zůstal nezměněný?

Můžeme použít, tzv. ratio test:

$$\frac{RHS_i}{t_{ij}} \leq \frac{RHS_k}{t_{kj}}$$

kde pro řady k, platí $t_{kj} > 0$. t_{ij} jsou proměnné ze základního výchozího řešení.

Tam, kde testová hodnota vyjde větší než 0, máme jistotu, že danou proměnou můžeme nahradit, aniž bychom dospěli k záporné pravé straně.

Uvádím 2 příklady, kdy po stisknutí tlačítka „RT“ byli vypočítané testy a doporučené proměnné na „výměnu“.

BV	x_1	x_2	x_3	x_4	x_5	RHS	CL	RT
x_3	2	1	1	0	0	40	☐	---
x_4	1	1	0	1	0	30	☐	30
x_5	1	0	0	0	1	15	☐	---
	☐	☐	☐	☒	☐		Pivot	Reset

Obrázek 3-12 - výsledek výp. mechanismu, kde je pro "klíčový" prvek doporučen prvek ve 4.sloupci a 2. řádku

3.2.3 Účelová funkce

Doposud jsme řešili pouze problémy spojené se základním řešením. v praxi se ale mnohem častěji setkáváme s případy, kdy je potřeba optimalizovat – tj. stanovit takové řešení, které optimálně splňuje zadané podmínky, které jsou vytvořeny, tzv. účelovou funkcí.

Účelová funkce z má následující tvar:

$$Z - c_1x_1 - \dots - c_nx_n = 0$$

a lze ji snadno přidat do celého modelu.

Písmenko c je zvoleno záměrně, jako zkratka pro náklady.(costs = náklady) Účelová funkce často vyjadřuje náklady na faktor, které je potřeba optimalizovat.

Během operací upravující výpočetní matici, pak provádíme úpravy i v účelové funkci z . Výpočetní mechanismus je tak upraven následujícím způsobem:

BV	x_1	x_2	x_3	x_4	x_5	RHS	CL	RT
x_3	1	0	1	-1	0	10	☐	
x_2	1	1	0	1	0	30	☐	
x_5	1	0	0	0	1	15	☐	
Z	-4	-3	0	0	0	0		
	☐	☐	☐	☐	☐		Pivot	Reset

Obrázek 3-13 - výpočetní mechanismus upravený o základní funkci z .

3.2.4 Optimalizace

Optimalizace je klíčovým faktorem, pro řešení úloh lineárního programování.

Spočívá v nalezení řešení, kdy je účelová funkce maximální nebo minimální. Abychom věděli, jaké volit klíčové prvky, je potřeba se držet následujících pravidel:

- pokud maximalizujeme, vybíráme takové nezákladní proměnné, které mají negativní (záporné) redukované náklady. (koeficient c v účelové funkci)
- pokud minimalizujeme, vybíráme takové nezákladní proměnné, které mají pozitivní (kladné) redukované náklady.

Při iteracích (tzn. úpravách daných matic) dle klíčových prvků, je potřeba „vědět, kdy přestat“, tj. kdy skončit, protože máme maximální nebo minimální hodnotu dané funkce. Projekt tutORial nám opět doporučuje následující pravidla:

- pokud maximalizujeme, ukončíme proces iterací, když všechny redukované náklady nezákladních proměnných jsou nezáporné. (tj. nulové nebo kladné)
- pokud minimalizujeme, ukončíme proces iterací, když všechny redukované náklady nezákladních proměnných jsou nekladné. (tj. nulové nebo záporné)

Při respektování výše uvedených pravidel a využití upraveného interaktivního výpočetního mechanismu, dostáváme pro základní rovnici:

$$z - 4x_1 - 3x_2 = 0$$

pro maximalizaci výsledek: $z + x_3 + x_2 = 100$,

BV	x_1	x_2	x_3	x_4	x_5	RHS	CL	RT
x_1	1		1	-1		10	☐	15
x_2		1	-1	2		20	☐	---
x_5			-1	1	1	5	☒	5
Z			1	2		100		max
CL	☐	☐	☐	☐	☒		Pivot	Reset

Obrázek 3-14 - maximalizace ve výp. mechanismu obsahujícím základní funkci z

a pro minimalizaci: $z - 4x_1 - 3x_2 = 0$, avšak se změněnými pravými stranami základního řešení.

BV	x_1	x_2	x_3	x_4	x_5	RHS	CL	RT
x3	2	1	1			40	☐	---
x4	1	1		1		30	☒	30
x5	1	0	0	0	1	15	☐	---
Z	-4	-3	0	0	0	0		min
CL	☐	☐	☐	☒	☐	Pivot	Reset	

Obrázek 3-15 - minimalizace ve výp. mechanismu obsahujícím účelovou funkci z

3.2.5 Speciální případy

„Violations“ neboli narušení základního modelu mohou být tři typů:

- záporný koeficient pravé strany ($b_i < 0$)
- rozhodující proměnná nemusí být nezáporná
- funkční podmínka, která není typu „ $\leq$ “

Ke všem narušením nám ale projekt tutORial uvádí, jak se s daným problémem vypořádat:

- nejsnadnější úprava; narušující proměnnou (celý řádek) vynásobíme číslem -1; pokud se v daném řádku vyskytuje nerovnost, je potřeba po vynásobení nerovnost „otočit“ opačným směrem
- u některých proměnných může nastat případ, kdy nechceme, aby byla nezáporná; v tomto případě se použije jednoduché matematické pravidlo, kdy každá neznámá (ať už kladná či záporná) může být vyjádřena jako rozdíl dvou jiných nezáporných proměnných; např. pokud $x = -4$, můžeme nahradit $x = v - w$, kde $v = 6$ a $w = 10$; tím nám do modelu mohou přibýt další proměnné
- mohou nastat dva případy:
 - omezení je typu: „ $\geq$ “: přepíšeme omezení s využitím rovnítka; např. rovnice $3x_1 + 2x_2 + 7x_3 \geq 20$ můžeme přepsat do tvaru:
$$3x_1 + 2x_2 + 7x_3 - x_4 = 20, (x_4 \geq 0)$$
 - omezení je typu: „ $=$ “: řešíme dočasným přidáním další proměnné do podmínky, tak abychom ji využili jako základní proměnnou; např. rovnici
$$x_1 + 3x_2 + 7x_3 - x_4 = 25$$
můžeme přepsat do tvaru:
$$x_1 + 3x_2 + 7x_3 - x_4 + x_5 = 25, (x_5 = 0)$$

z výše uvedeného je patrné, že model se může „zkomplikovat“ rozšířením o dodatečné proměnné. Existuje několik způsobů, jak se v tomto neztratit. Jednou z možností je dodatečné proměnné podtrhávat, druhá možnost je však mnohem více rozšířená. Jedná se o 2-fázovou simplexovou metodu.

3.2.6 Dvufázová simplexová metoda

Tato fáze není v projektu tutORial dopracována dokonce, proto uvedu jen stručnou charakteristiku dle jiného zdroje.

Pokud nejsou v úloze lineárního programování všechny omezující podmínky ve tvaru nerovnic „ $\leq$ “, není řešení této úlohy vůbec snadné a představuje celou I. fázi výpočtu. II. fáze se zabývá vlastní optimalizací účelové funkce.⁶

3.2.7 Simplexové řešitele

v této sekci jsou uvedeny další submoduly, které slouží jen jako výpočetní mechanismy.

Najdeme zde 4 výpočetní mechanismy, které jsou výpočetně podobné dříve uvedeným, výhodou je však jejich použitelnost pro „obsáhlé“ matice. (i 10x10) Jde o standardní formulář (standard form), obecný formulář (general form), aplikaci pro duální simplexovou metodu (Dual Simplex) a výpočetní mechanismus pro dopravní (zásobovací) problém v simplexové metodě.(transportation simplex)

Next
Transportation Module
Reset

Enter a basic feasible solution
(eg use the NW Corner Method)

i / j	1	2	3	4	5	s(i)	
1	8	8	2	4	6	13	13
2	4	9	9	3	2	20	20
3	6	5	9	6	5	9	9
d(j)	9	15	14	4	0	42	
	9	15	14	4	0		

Obrázek 3-16 - náhled výpočetního mechanismu pro dopravní problém, kde řádky jsou místa odbytu, zatímco sloupce značí místa poptávky

⁶ J. JABLONSKÝ: Operační výzkum, Professional publishing, 2002, str.61

3.3 Virtuální dualita

Je dalším tématem v rámci lineárního programování.

Bohužel není teoreticky rozpracován a v rámci projektu tutORial nabízí jen výpočetní mechanismus bez bližšího uvedení. To se předpokládá do budoucna.

Matrix Representation

opt $c'x$
s.t.
 $Ax = b$
 $x \geq 0$

Scalar Representation

opt $c_1x_1 + \dots + c_nx_n$
s.t.
 $a_{1,1}x_1 + \dots + a_{1,n}x_n = b_1$
.....
 $a_{m,1}x_1 + \dots + a_{m,n}x_n = b_m$
 $x_1, \dots, x_n \geq 0$

Opt = Min m = 2 ; n = 2

Go for it!

Obrázek 3-17 - náhled modulu simulující virtuální dualitu

3.4 Duální simplexová metoda a dopravní problém

Projekt tutORial v této sekci nabízí přímý odkaz na submodely již popsané v předchozí kapitole.

3.5 Maďarská metoda

Tento modul nabízí implementaci maďarského algoritmu při řešení tzv. přiřazovacího problému. Je poměrně složitý na pochopení bez patřičného teoretického uvedení do problému. v této práci se mu proto nebudu detailněji věnovat.

3.6 Zhodnocení modulů lineárního programování

Celou část týkající se Simplexové metody hodnotím jako velmi detailně propracovanou a prakticky velmi dobře využitelnou v rámci OR/MS disciplín.

Ostatní sekce v této části jsou v porovnání se Simplexovou metodou teprve „rozpracované“ a bylo by třeba je nadále rozvíjet.

Praktickou využitelnost těchto modulů vidím jen pro velmi dobře teoreticky připravené čtenáři, což je v rozporu s hlavním záměrem projektu tutORial, kladoucí si za cíl představení problémů OR/Ms širší veřejnosti.

4 Celočíselné lineární programování v rámci projektu tutORial

4.1 Obecné předpoklady a východiska⁷

Takové úlohy lineárního programování, které jsou doplněny o další požadavek, aby všechny nebo alespoň některé proměnné byly celá čísla se nazývají úlohy celočíselného lineárního programování. Použití celých čísel v metodách lineárního programování přináší další modelovací flexibilitu v rámci řešení a tím i rozšíření praktické využitelnosti.

4.2 Úloha o batohu (The Knapsack problem)

Modul v rámci celočíselného programování, který se zabývá Knapsack(batohovým) problémem.

Definice: Máme batoh o objemu v a n sad ($j=1,2,\dots,n$) položek. Každá sada obsahuje nekonečně mnoho identických položek, kde nekonečně mnoho kusů má stejnou váhu (w_j) a objem (v_j). Cílem je určit, kolik položek z každé sady x_j bychom měli umístit do daného batohu, tak abychom maximalizovali celkovou váhu daného prostoru – samozřejmě, aniž bychom překročili celkový objem V .

Existují dva pohledy, jak k danému problému přistupovat, každým z nich se zabývá jeden ze submodulů této části.

4.2.1 Neomezený problém

„Unbounded problem – branch and bound“, volně přeloženo jako neomezený problém – větvení a meze.

Jinak řečeno, tak jak jsem uvedla v definici, existuje nekonečně mnoho (neomezeně) položek v každé sadě. Matematicky lze problém popsat následovně:

⁷ Anderson, Sweeney, Williams: Introduction to management science, West publishing, 1994

$$\begin{aligned} z^* &:= \max z = w_1x_1 + w_2x_2 + \dots + w_nx_n \\ \text{s.t.} \quad &v_1x_1 + v_2x_2 + \dots + v_nx_n \leq V \\ &x_1, x_2, \dots, x_n \in \{0, 1, 2, 3, \dots\} \end{aligned}$$

tj. optimalizujeme celkovou váhu vyjádřenou účelovou funkcí z^* , za podmínek neomezenosti počtu kusů v n -té sadě a nepřekročení celkového objemu batohu.

v praxi se vyskytují další dvě obměny na tento problém:

- 1) bounded problem – tzn. ohraničený problém; počet položek v jednotlivé sadě je určen
- 2) 0-1 problem – položky mohou nabývat jen hodnot 0 a 1

Postup řešení je stejný, ne-li jednodušší jako u neomezené podmínky.

Proces řešení v rámci metody „BB“ je tento:

Krok 1: před-příprava: snažíme se snížit velikost problému vyřazením nevhodných položek anebo snížením velikosti batohu.(V) Zároveň se snažíme položky uspořádat sestupně podle jejich hustoty (w_j/v_j .)

Krok 2: zahájení: snažíme se najít vhodné řešení problému, tím že batoh zaplníme co největším množstvím položek typu 1, poté co nejvíce položkami typu 2, atd.

Krok 3: redukční tah: odstraníme jednu položku z batohu

Krok 4: Expanzivní tah: pokusíme se vylepšit současný výběr použitím položek z konce seznamu (těch nejmenších)

Proces končí ve chvíli, kdy už není možná žádná další redukce. (krok 2)

Abychom toto nemuseli počítat ručně, projekt tutORial nabízí výpočetní mechanismus, který dokáže problém spočítat.

Například zadání problému, kdy $n=3$, $V=10$, $w=(11,12,7)$, $v=(4,5,3)$ a jeho následný výpočet by vypadal následovně:

Volume, V = 10			
Stage, i	1	2	3
Pile, j	1	2	3
Weight, w(j)	11	12	7
Volume, v(j)	4	5	3
Densities, d(j)	2.75	2.4	2.333333
Decision, x(j)	1	0	2
Iteration :	6		
j :	3		
v :	6		
LB =	25		

Obrázek 4-18 - výpočetní mechanismus Knapsack problému (metoda BB)

Tzn., že použijeme jeden kus typu jedna a dva kusy typu 3, což nám nepřekročí celkový objem a zároveň maximalizuje váhu batohu na 25 jednotek.

4.2.2 Neomezený problém – dynamické programování

Druhý přístup k „neomezenému“ problému nám nabízí větší flexibilitu při řešení.

Matematicky by se vše dalo formulovat následujícím způsobem:

$$\begin{aligned}
 z^* &:= \text{opt } z = w_1x_1 + w_2x_2 + \dots + w_nx_n \\
 \text{s.t.} \quad &v_1x_1 + v_2x_2 + \dots + v_nx_n \leq V \\
 &x_1, x_2, \dots, x_n \in \{0, 1, 2, 3, \dots\}
 \end{aligned}$$

tj. celkovou váhu batohu můžeme i minimalizovat a objem přitom můžeme chtít roven přesně maximální hranici.

Tímto pak výpočetní modul dostává nové parametry na výběr a příklad z minulé kapitoly by vypadal následovně:

Solve	Help	Exit	Resize (n)	min	Constraint: =	Volume: 10																				
<table border="1"> <thead> <tr> <th>Pile</th> <th>1</th> <th>2</th> <th>3</th> </tr> </thead> <tbody> <tr> <td>Weights</td> <td>11</td> <td>12</td> <td>7</td> </tr> <tr> <td>Volumes</td> <td>4</td> <td>5</td> <td>3</td> </tr> <tr> <td>Decisions</td> <td>0</td> <td>2</td> <td>0</td> </tr> <tr> <td>z*</td> <td>?</td> <td colspan="2">tutORial</td> </tr> </tbody> </table>							Pile	1	2	3	Weights	11	12	7	Volumes	4	5	3	Decisions	0	2	0	z*	?	tutORial	
Pile	1	2	3																							
Weights	11	12	7																							
Volumes	4	5	3																							
Decisions	0	2	0																							
z*	?	tutORial																								

Obrázek 4-19 - výpočetní mechanismus Knapsack problému (metoda DP)

tj. minimalizujeme celkovou váhu při přesném(maximálním) objemu.

4.3 Gomoryho metoda řezných nadrovin

Tato metoda není v projektu tutORial teoreticky rozebrána a autor uvádí poznámku, že výpočetní mechanismus je jen pro odborníky, kteří vědí co mají dělat. Vzhledem k nedostatku materiálů o této metodě, jsem se do hlubšího prozkoumávání tohoto modelu nepouštěla.

Náhled na výpočetní mechanismus však uvádím v příloze.

4.4 Úloha osmi dam

Název tohoto modulu, „Royal optimization“, mě velmi zaujal a představovala jsem si, jak asi vypadá královská optimalizace?

Odpověď byla prostá a možná i předvídatelná: problém královské optimalizace je spojen s královskou hrou – šachy. Jde o to, rozmístit na šachovnici N-královen, tak aby jedna druhou „nevyhodila“.

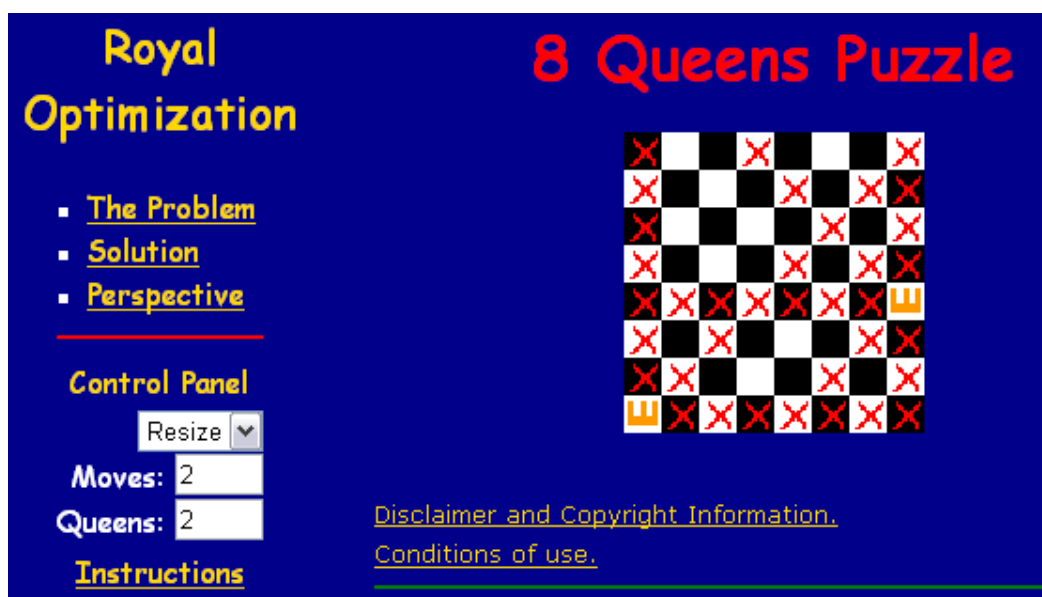
Královný je potřeba rozmístit na šachovnici o $N \times N$ polích.

Model nabízí i netradiční přístup k problému. Klade otázku, aby se čtenáři pokusili problém formulovat z hlediska OR/MS.

Zadání je následující:

- formulovat problém pro $n=30$, tj. 30 královen
- problém zároveň vyřešte jednou z metod LP a použijte výpočetní mechanismy v rámci projektu tutORial

Náhled JAVA aplikace královské hry pro 8 královen, kde jsem již umístila 2 královny na pole (1,1) a (4,8).



Obrázek 4-20 - Náhled JAVA aplikace královské hry pro 8 královen

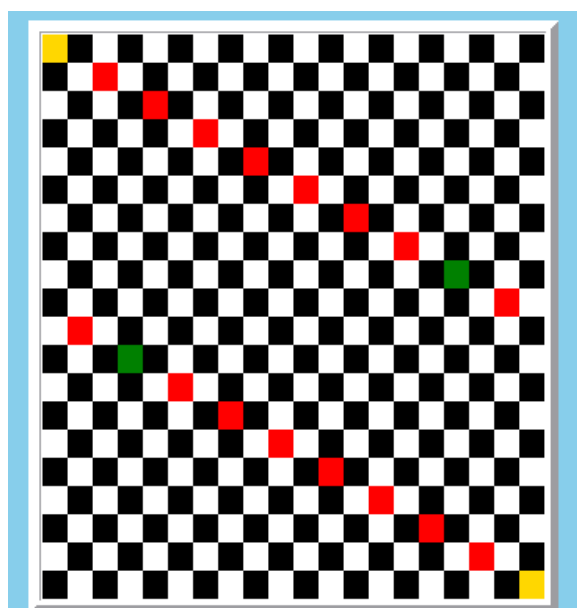
4.5 Úloha osmi dam - nově

Část „New royal optimization“ nabízí rozšíření a nový pohled na královskou optimalizaci. Problém je stejný, tj. rozmístění N-královen na šachovnicové pole.

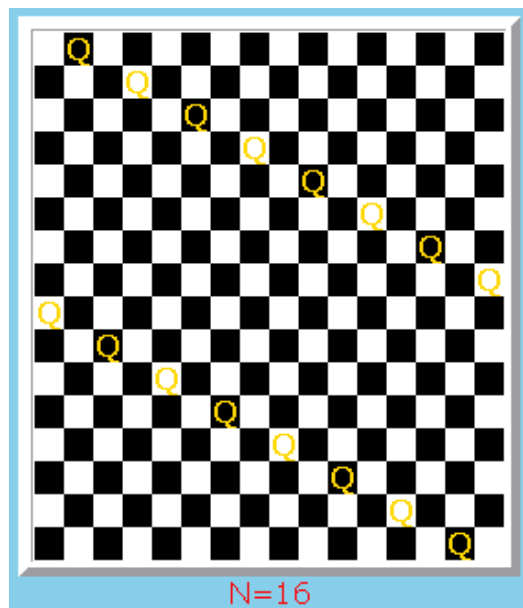
Na rozdíl od modulu předchozího, ale nabízí i návod, jak daný problém vyřešit.

Princip lze shrnout do dvou kroků:

1. Vyjádříme N jako rovnici $N = 6k + m$, kde k je nejvyšší možné celé číslo, které splňuje podmínku $6k \leq N$.
2. Použijeme odpovídající přístup dle hodnoty m :
 - a. pokud m nabývá hodnot $\{0,1,4,5\}$ použijeme mimo-diagonální přístup. (viz obrázek 4-5)
 - b. pokud je $m = 2$, použijeme diagonální přístup (viz obrázek 4-4)
 - c. pokud je $m = 3$, použijeme řešení pro $m=2$ v sekci $(N-1)*(N-1)$ a poté doplníme královnu do buňky (N,N)



Obrázek 4-21 - diagonální přístup v RO



Obrázek 4-22 – mimo-diagonální přístup v RO

Přehled náhledů pro různá N , stejně tak jako náhled hrací plochy, připravené pro umístění královen uvádím v příloze č.2.

4.6 8 snadných kusů (8 Easy pieces)

Poslední část kapitoly celočíselného programování v rámci projektu tutORial. Funguje na obdobném principu, jako „královská optimalizace“, tj. celý model je interaktivní hra, kde máme vymyslet postup v rámci OR/MS.

Opět se jedná o známou hru, o které se autoři domnívají, že by se měla stát součástí tradičních postupů v rámci OR/MS.

„Zaprvé, tato hra se svým prostředím má velkou šanci stát integrovanou součástí OR/MS prostředí. Za druhé je výzvou pro „tradiční“ OR/MS. Je tomu tak proto, že spojuje programování založené na podmínkách a celočíselné programování.“

Ve hře je potřeba seřadit políčka, tak jak byla seřazena před zamícháním.



Obrázek 4-23 - náhled hry "8 Easy Pieces"

4.7 Zhodnocení modulů celočíselného programování

v porovnání s moduly předchozí kapitoly jsou modely celočíselného programování na nižší úrovni rozpracovanosti. Nejlépe a nejdetailněji je rozpracovaná část první zabývající se problémem větvení a hranic. Tuto část hodnotím i velmi dobře z hlediska praktické využitelnosti.

Ostatní části v této sekci opět nejsou až tak teoreticky rozpracované, což může být jejich negativum. Naopak bych ale vyzdvihla nápadité hry, které mají ambice se stát detailně rozpracovanými problémy OR/MS.

5 Závěr

Hlavním cílem mé bakalářské práce bylo představení projektu tutORial, který má za cíl seznámit širší veřejnost s problémy OR/MS a zhodnotit jeho využitelnost pro praktickou výuku.

Práci jsem rozdělila celkem do pěti oddílů. Každý z oddílů byl věnovaný konkrétní části OR/MS disciplín. v prvním z těchto stěžejních oddílů jsem stručně představila a zhodnotila všechny moduly uvedené v rámci projektu tutORial. v dalších dvou oddílech jsem se pak věnovala konkrétním problémům v rámci Lineárního programování a celočíselného programování. Obě části jsem zhodnotila z hlediska praktické využitelnosti.

Podle tohoto hodnocení v rámci jednotlivých kapitol byla až na výjimky prokázána velmi dobrá praktická využitelnost většiny modulů uvedených v rámci projektu tutORial.

v přehledu všech modelů jsem ocenila velmi dobrou teoretickou i praktickou zpracovanost a využitelnost některých modelů i v jiných disciplínách než je OR/MS.

v části věnované lineárnímu programování bych vyzdvihla velmi podrobnou část věnovanou simplexové metodě, která přináší i výbornou praktickou využitelnost.

v poslední části jsem zmínila velký potenciál pro další rozpracovávání.

Dalším z cílů projektu tutORial je rozvinout komplexní web zabývající se komplexními online problémy disciplín OR/MS, což se v případě projektu tutORial podařilo.

Použitá literatura

Knihy a učebnice

1. Anderson, D.R., Sweeney, D.J., Williams, T.A.: Introduction to management science, West publishing, 1994
2. Eiselt, A.: Integer programming and network models, Berlin, Springer, 2000
3. Chvátal, A.: Linear programming, W. H. Freeman, New York, 1983
4. Jablonský, J.: Operační výzkum, Professional publishing, 2002
5. Lagová, M.: Lineární modely, Oeconomica, Praha, 2004
6. Lauber, L.: Programy pro matematické modelování, VŠE, Praha, 1993
7. Ross, M.: Operational research '72, American elsevier publishing, 1973
8. Synek, M. Jak psát diplomové a jiné písemné práce, skripta VŠE, Praha 2002

Internetové stránky

1. www.ifors.ms.unimelb.edu.au/tutorial/
2. <http://www.ifors.org/>
3. <http://nb.vse.cz/csov/>

Seznam obrázků

OBRÁZEK 2-1 - NÁHLED PRAKTICKÉ APLIKACE "DIE HARD AT THE PUB".....	8
OBRÁZEK 2-2 - PRAKTICKÁ UKÁZKA PROBLÉMU "NEJKRATŠÍ CESTA".....	9
OBRÁZEK 2-3 - DIAGRAMY A MATICE MODELU "NEJKRATŠÍ CESTA".....	9
OBRÁZEK 2-4 - NÁHLED NA TUTORIAL NÁVOD PRO TOPOLOGICKÉ TRÍDĚNÍ.....	10
OBRÁZEK 2-5 - HL. OKNO PROGRAMU, KTERÝ DOKÁŽE PROVÁDĚT ŘÁDKOVÉ ÚPRAVY MATIC.....	11
OBRÁZEK 2-6 - HL. OKNO PROGRAMU, KTERÝ VYTVÁŘÍ INVERZNÍ MATICI.....	12
OBRÁZEK 2-7 - HL. OKNO PROGRAMU, KTERÝ POČÍTÁ DETERMINANT.....	12
OBRÁZEK 2-8 - NÁHLED NA APLIKACI POMÁHAJÍCÍ PŘI VÝPOČTECH PROBLÉMU HROMADNÉ OBSLUHY.....	14
OBRÁZEK 3-9 - VÝCHOZÍ SOUSTAVA ROVNIC A JEJÍ ZOBRAZENÍ V PROJEKTU TUTORIAL.....	16
OBRÁZEK 3-10 - VÝCHOZÍ ZADÁNÍ DO VÝPOČETNÍHO MECHANISMU GAUSSOVA ELIMINAČNÍHO PROCESU; URČENÍ KLÍČOVÉHO SLOUPCE A KLÍČOVÉHO ŘÁDKU.....	16
OBRÁZEK 3-11 - VÝSLEDEK VE VÝPOČETNÍM MECHANISMU GAUSSOVA ELIMINAČNÍHO PROCESU; BV – OZNAČENY ZÁKLADNÍ PROMĚNNÉ; RHS – PRAVÁ STRANA	17
OBRÁZEK 3-12 - VÝSLEDEK VÝP. MECHANISMU, KDE JE PRO "KLÍČOVÝ" PRVEK DOPORUČEN PRVEK VE 4.SLOUPCI A 2. ŘÁDKU.....	18
OBRÁZEK 3-13 - VÝPOČETNÍ MECHANISMUS UPRAVENÝ O ZÁKLADNÍ FUNKCI Z.....	18
OBRÁZEK 3-14 - MAXIMALIZACE VE VÝP. MECHANISMU OBSAHUJÍCÍM ZÁKLADNÍ FUNKCI Z.....	19
OBRÁZEK 3-15 - MINIMALIZACE VE VÝP. MECHANISMU OBSAHUJÍCÍM ÚČELOVOU FUNKCI Z.....	20
OBRÁZEK 3-16 - NÁHLED VÝPOČETNÍHO MECHANISMU PRO DOPRAVNÍ PROBLÉM, KDE ŘÁDKY JSOU MÍSTA ODBYTU, ZATÍMCO SLOUPCE ZNAČÍ MÍSTA POPTÁVKY.....	21
OBRÁZEK 3-17 - NÁHLED MODULU SIMULUJÍCÍ VIRTUÁLNÍ DUALITU.....	22
OBRÁZEK 4-18 - VÝPOČETNÍ MECHANISMUS KNAPSACK PROBLÉMU (METODA BB)	26
OBRÁZEK 4-19 - VÝPOČETNÍ MECHANISMUS KNAPSACK PROBLÉMU (METODA DP)	26

OBRÁZEK 4-20 - NÁHLED JAVA APLIKACE KRÁLOVSKÉ HRY PRO 8 KRÁLOVEN	28
OBRÁZEK 4-21 - DIAGONÁLNÍ PŘÍSTUP V RO OBRÁZEK 4-22 – MIMO-DIAGONÁLNÍ PŘÍSTUP V RO.....	29
OBRÁZEK 4-21 - DIAGONÁLNÍ PŘÍSTUP V RO OBRÁZEK 4-22 – MIMO-DIAGONÁLNÍ PŘÍSTUP V RO.....	29
OBRÁZEK 4-23 - NÁHLED HRY "8 EASY PIECES"	30

Seznam příloh

PŘÍLOHA Č. 1 – NÁHLED APLIKACE ŘEŠÍCÍ LP METODOU GOMORYHO ŘEZNÝCH NADROVIN

PŘÍLOHA Č. 2 – PŘEHLED NÁHLEDŮ „KRÁLOVSKÉ OPTIMALIZACE“ PRO RŮZNÁ N , A NÁHLED HRACÍ PLOCHY, PŘIPRAVENÉ PRO UMÍSTOVÁNÍ KRÁLOVEN

PŘÍLOHA Č. 3 – NÁHLED ÚVODNÍ STRANY PROJEKTU TUTORIAL

Příloha č. 1 – Náhled aplikace řešící LP metodou Gomoryho řezných nadrovin

Go!
Randomize
Fasten your seatbelts!
Density

Resize
m =
n =
Width

C							
	[1]	[2]	[3]	[4]	[5]	= v	b
[1]						= v	[1]
[2]						= v	[2]
[3]						= v	[3]

$x_j \geq 0$

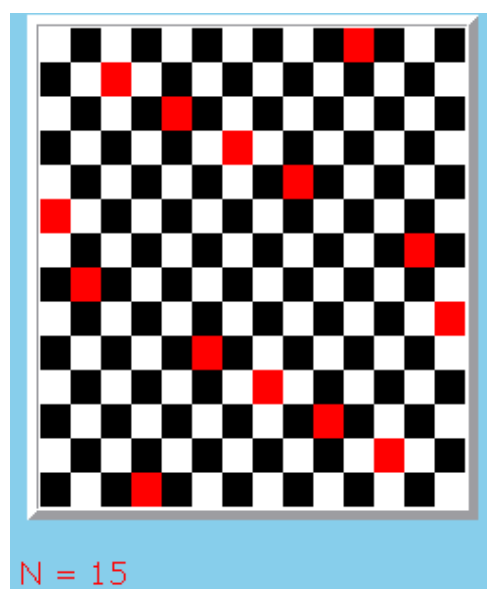
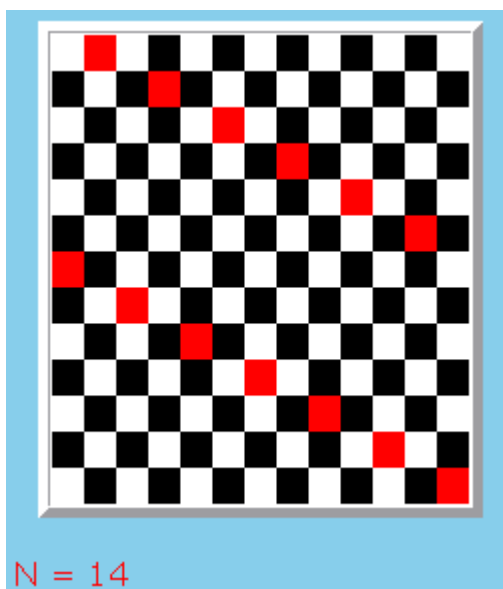
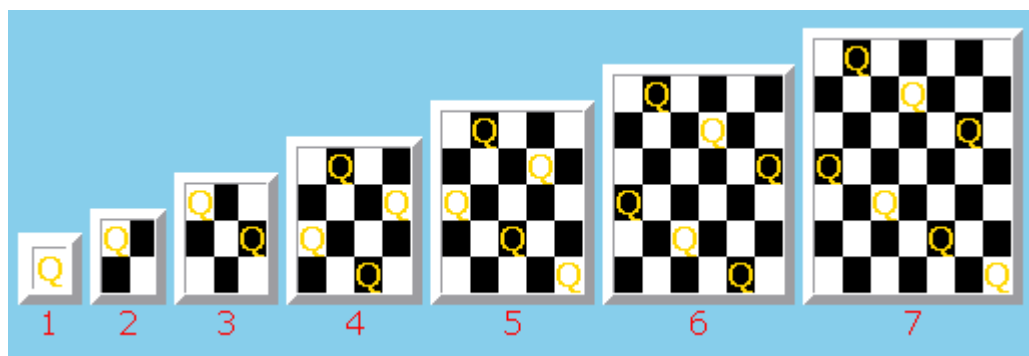
☒
☒
☒
☒
☒

Le Simplex

Instructions

- Make sure that the specified **size** of the problem is the one you need. If this is not so, use the **Resize** button to resize the problem. Do this **before** you input the data.
- Use the **Width** button to change the width of the columns. Set the width of the columns **before** you fill-in the form.
- Uncheck the checkboxes corresponding to **URS variables** (variables that are not required to satisfy the non-negativity constraint $x_j \geq 0$).

Příloha č. 2 – Přehled náhledů „Královské optimalizace“ pro různá N, a náhled hrací plochy, připravené pro umístění královen



Příloha č. 3 – náhled úvodní strany projektu tutORial

