ROZVOZNÍ ÚLOHY. ZAVEDENÍ ČASOVÝCH OKEN

Nejprve k pojmům používaným v okružních a rozvozních úlohách:

HAMILTONŮV CYKLUS je typ cesty, kdy každý uzel grafu navštívíme právě jednou a pak se vrátíme do výchozího místa. V úloze obchodního cestujícího hledáme Hamiltonův cyklus s minimálním součtem ohodnocení hran.

EULERŮV TAH JE TAH, KTERÝ OBSAHUJE KAŽDOU HRANU PRÁVĚ JEDNOU.

EULERŮV CYKLUS JE TAKOVÝ CYKLUS, KDY KAŽDOU HRANOU V GRAFU PROJDEME PRÁVĚ JEDNOU, A PAK SE VRÁTÍME DO VÝCHOZÍHO MÍSTA. SETKÁVÁME SE S NÍM V ÚLOZE ČÍNSKÉHO LISTONOŠE.

FLEURYHO ALGORITMUS je algoritmus pro hledání Eulerova cyklu.

Ve STATICKÝCH ÚLOHÁCH známe všechny požadavky předem, v DYNAMICKÝCH ÚLOHÁCH po výjezdu vozidel přicházejí další požadavky.

V OKRUŽNÍCH ÚLOHÁCH neuvažujeme velikost požadavků, zatímco v ROZVOZNÍCH ÚLOHÁCH ano, takže zde hraje roli i kapacita vozidla.

ROZVOZNÍ ÚLOHA

Uvažujme situaci, kdy máme nějaký výchozí uzel a zákazníky v uzlech i = 2,3…n, kteří mají určité požadavky o velikosti q_i. Naše vozidlo má kapacitu V, přičemž součet požadavků zákazníků převyšuje kapacitu vozidla. Známe opět vzdálenosti mezi uzly i a j, které značíme c_{ij}. Naším cílem je uspokojit požadavky všech zákazníků a vrátit se do výchozího místa tak, aby součet délek všech tras byl minimální a abychom zároveň nepřekročili kapacitu vozidla. Na rozdíl od úlohy obchodního cestujícího tak může výsledné řešení obsahovat více cyklů, ale v každém z nich musí být obsažen výchozí uzel.

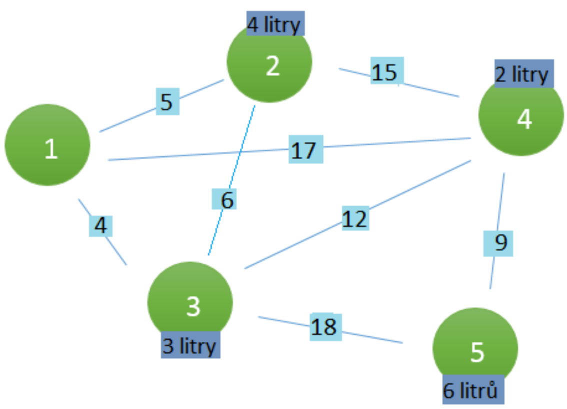
Jde sice o rozvozní úlohu, ale modelujeme ji jako svozní problém: v každém místě přidáme do vozidla takový náklad, který odpovídá výši požadavku právě navštíveného zákazníka, a to až do naplnění kapacity vozidla. Z hlediska optimalizace jde o totéž, jako kdybychom vyjeli s plným vozidlem a to postupně vykládali.

Zavedeme binární proměnnou x_{ij}, která se bude rovnat 1 v případě, že vozidlo pojede z i do j, jinak se bude rovnat 0. Dále zavedeme proměnnou u_i, která označuje výši nákladu v uzlu i. Model formulujeme následovně:

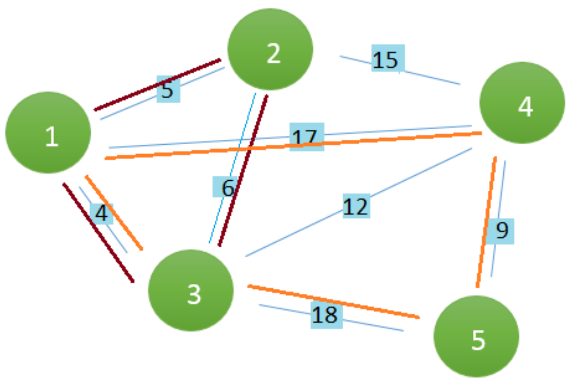
$z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \rightarrow min$	*Snažíme se, aby součet vzdáleností byl minimální.*
$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 2,3 \dots n$	*(1) Pro každý uzel musí platit, že z něj právě jednou vyjedeme. Výjimkou je první uzel, kam si musíme jezdit doplnit zboží,*
$\sum_{i=1}^{n} x_{ij} = 1 \quad j = 2,3 \dots n$	*(2)Pro každý uzel musí platit, že do něj právě jednou vjedeme. Výjimkou je první uzel, kam si musíme jezdit doplnit zboží.*
$x_{ij} \in \{0,1\} \quad i,j = 1,2 \dots n$	*(3) Pojedeme z uzlu i do uzlu j?*
$u_1 = 0$	*(4) Náklad při výjezdu z prvního uzlu nastavíme na 0.*
$q_i \leq u_i \leq V \quad i = 2,3 \dots n$	*(5) Pro všechny uzly (kromě prvního) musí platit, že náklad po navštívení tohoto uzlu, je alespoň roven požadavku tohoto uzlu. Přitom ale náklad nebude větší, než dovolí kapacita vozidla.*
$u_i + q_j - V(1 - x_{ij}) \leq u_j \quad i = 1,2 \dots n, \; j = 2,3 \dots n, i \neq j$	*(6) Pokud nejedeme z i do j, pak bude $x_{ij} = 0$ a podmínka bude splněna vždy. Pokud ale pojedeme z i do j, pak musí platit, že náklad v uzlu i navýšený o požadavek uzlu j nemůže být větší než náklad v uzlu j.*

```
!Uvažujme graf níže. Anička bydlí s rodiči na farmě a každý den rozváží čtyřem
sousedům mléko. Tito sousedi si přejí, aby jim dovezla denně 4, 3, 6 a 2 litry
mléka. Anička ale najednou uveze pouze 9 litrů mléka. Jak si má naplánovat cestu,
aby ujetá vzdálenost byla co nejmenší?
;
model:
sets:
uzel/1..5/:naklad, pozadavky;
cesta(uzel,uzel):naklady,x;
endsets

data:
naklady =
0  5  4 16 22
5  0  6 15 24
4  6  0 12 18
16 15 12 0 9
22 24 18 9 0;

pozadavky = 0 4 3 6 2;
kapacita = 9;
enddata

min = @sum(cesta: naklady*x);

@for(uzel(i)|i#NE#1: @sum(uzel(j): x(i,j)) = 1);!do každého uzlu kromě prvního
jednou vjedeme a jednou z něj vyjedeme;
@for(uzel(j)|j#NE#1: @sum(uzel(i): x(i,j)) = 1);

@for(cesta(i,j)|j#NE#1#AND#i#NE#j: naklad(i) + pozadavky(j) - 9*(1-x(i,j))
<=naklad(j));!naklad pri vyjezdu do i plus pozadavek uzlu j (= to, co v nem
nalozime) musi v pripade ze jedeme z i do j byt mensi nez naklad v uzlu j, jinak
plati vzdy;
@for(cesta: @bin(x));
@for(cesta(i,j)|i#EQ#j: x(i,j) = 0);
@for(uzel(i)|i#NE#1: naklad(i) <= kapacita);
@for(uzel(i)|i#NE#1: naklad(i) >= pozadavky(i));!když má uzel požadavek, abychom do
něj u něj naložili 5 ks, pak nemůžeme naložit napriklad jen 4 a vrátit se;

end
```

Zadání



Řešení



Stejně jako u obchodního cestujícího existuje i rozvozní úloha s časovými okny, ve které musí být každý zákazník navštíven jen v určitém časovém intervalu, tedy v okně <a_i,b_i>. Kromě vzdálenosti mezi uzly tak musíme znát jednak dobu přejezdu mezi uzly d_{ij}, jednak dobu nakládky u *i*-tého zákazníka S_i.

K modelu pak přidáme podmínku:

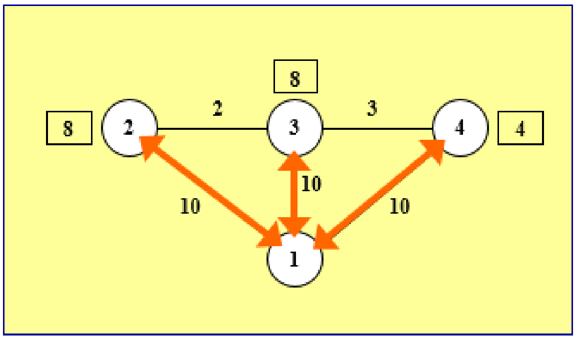
$t_1 = 0$		*(1) Dobu návštěvy zákazníka nastavíme na 0.*
$a_i \le t_i \le b_i$	$i = 2,3 \ldots n$	*(2) Pro všechny uzly (kromě prvního) musí platit, že čas, ve kterém jej navštívíme, spadá do požadovaného časového okna.*
$t_i + S_i + d_{ij} - M(1 - x_{ij}) \le t_j$	$i = 1,2 \ldots n$ $j = 2,3 \ldots n$ $i \neq j$	*(3) M je nějaké velké číslo. Pokud nejedeme z i do j, pak bude $x_{ij} = 0$ a podmínka bude splněna vždy. Pokud ale pojedeme z i do j, pak musí platit, že uzel j nenavštívíme dříve, než kolik činí čas návštěvy uzlu i plus doba nakládky v uzlu i plus doba přejezdu z i do j.*

Existují i další typy rozvozních úloh.

Máme K VOZIDEL S RŮZNOU KAPACITOU, kde V_k je kapacita *k*-tého vozidla. Pak rozlišujeme trasy podle vozidel. Pak pracujeme s proměnnými se třemi indexy. x_{ij}^k se rovná 1, pokud *k*-té vozidlo pojede z uzlu *i* do uzlu *j*.

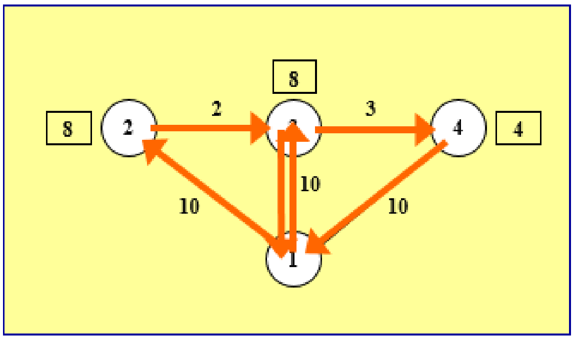
Existují úlohy s DĚLENOU DODÁVKOU. Pokud je mezi uzly takový, jehož požadavek překračuje kapacitu vozidla, pak je nutné dělit dodávku. Ale i pokud mezi uzly žádný takový není, může být výhodnější dodávku rozdělit, jak je patrné například z následujícího obrázku.

Řešení s nedělenou dodávkou



Celková vzdálenost = 60

Řešení s dělenou dodávkou



Celková vzdálenost = 45.

`Zdroj: Ing. Jan Fábry, Ph.D.: 4EK314 Diskrétní modely`

ZDROJE:

Ing. J. Fábry, Ph.D.: přednášky 4EK314 Diskrétní modely, 2011.