Poznámky a popis vzniku base-line algoritmu a návrhy možného řešení

Logistika ve skladu

Skladová logistika je velmi komplexní obor. Její složitost a propracovanost se obvykle odvíjí od zkušeností, které mají dané firmy, její velikostí a počtem produktů, které je třeba uskladnit až po finanční možnosti, které lze investovat do organizace a fungování skladu. Fungování skladu, u kterého uvažujeme propojení s elektornickým systémem firmy, můžeme rozdělit do čtyř základních úrovní:

1. Malý sklad s nízkým počtem položek. Zaměstnanci se umí orientovat po paměti a žádné skladové adresy nejsou potřeba. Zboží je organizováno v některých případech nejvýše do nějákých logických celků, popřípadě můžem být popsáno. Propojení zboží vedeného v systému firmy a jeho uložení na skladě neexistuje.
2. Sklad větších rozměrů se středním počtem zboží - je v něm přinejmenším obtížné orientovat se po paměti či s nepřenými označeními lokací jednotlivých položek. Proto taková firma velmi často volí implementaci WMS (Warehouse Management System). Ve skladu se zvolí adresový systém (způsob organizace a konvence zápisu adres). Ten se potom využije způsobem, že každá položka již má v elektronickém systému vedenu i její adresu na skladu, kde se nachází. Zaměstnanci ovšem stále nepoužívají elektronická zařízení.
3. Velký sklad s vysokým počtem zboží - ačkoli již má implementovaný WMS a tudíž je organizován podle adres, zaměstnanci ve většině případů používají tužku a papír během své práce, která sestává většinou z činností
    - během příjmu zboží do skladu - "vezmi položku a zařaď ji na sklad"
    - během exportu zboží ze skladu - "najdi položku na skladu a přines" přičemž je neustále nutné veškeré akce nejprve provést a následně je i registrovat do systému, což bývá náchylné k chybám. Proto je zavedeno používání elektroniky pro zjednodušení výše nastíněné práce. Tímto je zajištěno, že změny týkající se zboží, které provede pracovník, jsou v reálném čase provedeny i v elektronickém systému.
      Používaná zařízení (velmi často čtečky) navíc mohou sloužit i pro zrychlování práce, například možností načítat kódy požadovaného zboží právě díky použití čtečky.
4. Skladové prostory firem s miliardovými obraty - zde již je práce velmi efektivně řízena, nicméně začíná se utvářet prostor pro použití automatizace ve formě robotů. Jejich zavedení brání ve většině případů nemožnost finanční návratnosti, právě kvůli malému obratu. Robot s sebou přináší výhodu jednorázové investice a ušetření množství lidí, které ovšem musí na druhou stranu být dostatečně velké na to, aby byl robot v - pro firmu únosné - době splacen. Příkladem takové automatizace může být využití paletových či balících robotů.

Logistické problémy

Logistika je obor, kde se ve velkém množství uplatňují optimalizační procesy, které mají sloužit pro zrychlení procesů, úsporu zaměstnanců či zvýšení produktivity na skladech. Mnoho z problémů jsou již známé z

historie - patří mezi ně slavný *Problém obchodního cestujícího* či *Problém čínského listonoše*. Některé jsou novodobého rázu - například *Bin packing problem* nebo *Vehicle routing problem*.

- **Problém obchodního cestujícího (známý jako TSP - Travelling Salesman Problem)** je prolémem, který řeší průchod skrz graf tak, aby žádný vrchol nebyl navštíven vícekrát (konkrténě se jednalo o obchodníka, který má navštívit všechna na seznamu tak, aby urazil co nejkratší vzdálenost a zároveň nenavštívil žádné město víc než jednou). Patří do kategorie NP-úplných problémů a tedy zatím nemá řešení pro libovolný počet uzlů. Aktuálně je tento problém řešen heuristikami, kterými je například algoritmus *Nejblížšího souseda (Nearest neighbour)*, či použitím genetických algoritmů.

  Tento problém úzce souvisí s problémem Hamiltonova cyklu - to je cyklus obsahující takovou cestu, na které lze všechny uzly navštívit právě jednou a navrátit se přitom do počátečního vrcholu. (cit. Wróblewski P. Algoritmy)*Pokud najdeme takový (tj. Hamiltonovský) cyklus s minimální sumou jeho hran, vyřešíme slavný problém obchodního cestujícího.*

- **Problém čínského listonoše (známý jako Chinese postman Problem)** je jakýmsi protějškem problému TSP - zabýváme se totiž při něm hledáním způsobu, jak navštívit všechny hrany daného grafu, a to tak, aby každá byla navštívena v ideálním případě jedenkrát - to je sice předem známo (takový graf by totiž musel být Eulerovským - tedy má všechny uzly sudého stupně a tudíž existuje uzavřený tah obsahující všechny hrany), nicméně je cílem tohoto dosáhnout.

- **Bin packing problem** opět spadá pod NP-úplné problémy, nicméně existuje řada heuristik pro jeho výpočet. Má mnoho podob, jejichž cílem je vždy poskládání N předmětů do prostoru (boxu) o dané kapacitě K, či rozměrech X,Y,Z. Možnosti řešení jsou rozděleny od jednodušších ke složitějším. Z povahy problému jistě vidíme, že největší složitost budou mít algoritmy řešící druhou variantu, tedy skládání objektů do prostoru dle rozměrů.

  - Nejdříve se zaměřme na první variantu. Každý z N předmětů je definován veličinou X, X e R. Dále mějme P boxů, všechny o stejné maximální kapacitě (odpovídající veličině charakterizující předměty) K, K e R. Uveďme, že je nezbytnou podmínkou aby pro každé X náležející N platilo, že X(N) < K. Cílem je naplnit nejmenší možný počet boxů. **Tento typ problému zároveň využíváme v základním řešení bakalářské práce jako zjednodušení výpočtu s reálnými objemy**. Uveďme dva základní algoritmy:
    - First fit - v cyklu bereme po řadě či náhodně předměty N a vkládáme je vždy zleva do boxů (seřazených zleva doprava) takto:
      1. Po řadě bereme jednotlivé dosud nezařazené objekty a provedeme následující:
      2. Pro všechny dosud existující boxy po řadě provedeme:
         1. Pokud se objekt do daného boxu vejde (podle charakteristické veličiny), potom do něj objekt přidáme a bereme další objekt
         2. Jinak bereme zkoušíme další box. Pokračujeme krokem 2.
      3. Vytvoříme nový box a vložíme do něj aktuální objekt. Pokračujeme krokem 2.
    - Next fit - je nejjednodušší heuristikou pro vytvoření boxů naplněných objekty. Jeho průběh je zjednodušen oproti algoritmu First Fit pouze tím, že daný objekt je vyzkoušen pouze na posledním vytvořeném boxu, pokud se tam nevejde, tak je vytvořen pro něj box další.

Problémy a jejich složitost

Pro výběr vhodného algotimu, či vícero algoritmů se musíme seznámit s hodnotícími kritérii.

Hodnocení kvality algoritmů se odvíjí od dvou základních bodů - časové a prostorové složitosti. Tyto dva parametry jsou ovlivňovány velikostí vstupních dat. Uveďme, že zatímco význam prostorové složitosti je s narůstající kapacitou výpočetních pamětí (a s tím i klesáním ceny za GB) umenšován, význam časové složitosti je obrovský, vezmeme-li v úvahu algoritmy, které mají odhadovaný čas nalezení řešení v řádu staletí. Časová složitost se nejčastěji odhadnuta jako součet časů (považovaných za konstantní) jednotlivých elementárních operací. Vždyc nás zajímá nejhorší možný scénář.

Příliš náročné algoritmy (viz //TODO reference) jsou řešeny pomocí tzv. heuristik - algoritmů, které sice nezaručují nalezení optimálního řešení, zato umožňují najít vyhovující (podle předem stanoveného kritéria) řešení v polynomickém čase - takovém, kde existuje funkce představující horní ohraničení složitosti takového algoritmu pro daný počet vstupů //TODO math. T(n) = O(n^k) pro konstantu k>0.

Asymptotická složitost

U algoritmů zjišťujeme tzv. asymptotickou složitost - ta se vyjadřuje jako porovnáním algoritmu s jistou funkcí pro N limitně se blížící nekonečnu. Podle ní si dokážeme představit, jak se bude algoritmus chovat pro N vstupů, přesněji - jak dlouho jemu bude trvat výpočet.

- Omikron - horní hranice chování
- Omega - dolní hranice chování
- Theta - třída chování

Úvod do teorie grafů

Principy optimalizace

Úvod do problematiky řešené úlohy

- V systému je provedeno N objednávek, po čase T (odpovídající tomu, že systém vyhodnotí dostatečnou naplněnost, či podle jiného kritéria (pro nás zde nepodstatné) dojde k `zavlnění` - v této fázi jsou vytvořeny **dávky** (dále **HD** - Hromadný Dodací list - toto označení zastupuje jednoho zákazníka a zjednodušení pojmu dávka) a jednotlivým produktům je **přidělena adresa**; prodejní doklady na **stejného zákazníka a stejnou dodací adresu** jsou spojeny do jednoho **HD**. Výsledná data budou odeslána pro výpočet algoritmu, jakožto výsledku této práce.
- *Aktuálně systém pokračuje rozřazením HD do vozíků tak, že zohledňuje pouze objem, počet přihrádek ve vozíku (9) a kritérium nedělitelnosti zákazníka. Lidé pak chodí s vozíkem podle fixně řazeného seznamu adres.*
- Vstupem algoritmu (přesněji výpočtu) jsou položky (vázané na zákazníky), které se mají vychystat - *"jeden řádek - jedno místo ve skladu - 1+ kusů zboží - 1+ přepravek na vozíku"*
- Položky jsou ve formátu tabulky, jejíž sloupce jsou typů vypsaných níže (potenciální hodnoty jsou u nich pro lepší pochopení rovněž vypsány)

    - HD (Alza Praha 3, CZC Opava, COMFOR Poruba Hlavní třída,...)
    - Položka (Lenovo IdeaPad AX, Apple MacBook Pro 10, ThinkPad T150,...)
    - Počet kusů produktu (1, 5, 20,...)
    - Adresa (G000159545, PO10494564,...) určující místo na skladě
- K těmto řádkům (položkám) jsou dále k dispozici informace o produktech (viz snímek)

	ID	Product	Unit	PackageType	ContentQty	X	Y	Z	Weight
1	1	875	Car	BA	20	47.4	22.6	38.1	10.582
2	2	875	Unit	BA	1	18.9	8.5	9.2	0.488
3	3	875	Car	BA	20	45	22	43	9.24
4	4	2088	Unit	BA	1	40	11	26	0.95

    - Tyto informace potřebuji především pro výpočet objemu, abych následně věděl jak to vyskládat do vozíku a kolik přepravek bude třeba.
- **Pravidla**
  Pravidla zde napsaná je nutno dodržet, aby nedošlo ve skladu k nekonzistentním situacím.
    - Jeden zákazník (objednávky od jednoho zákazník na stejné adrese spojené do HD) je nedělitelný mezi vozíky
    - Vozík má devět přepravek
        - Rozměry přepravky jsou 28x42x18 cm
    - Zboží mající větší rozměry, než se vejde do přepravky musí být stejně vychystáno
    - Jedna přepravka může obsahovat zboží pouze od jednoho zákazníka (ten samozřejmě může být ve více přepravkách)
        - Jedna přepravka může obsahovat zboží od *více zákazníků*, pokud každé zboží je položka o jednom kusu
- **Cíl**

  ```
  Naskládat co nejefektivněji HD do vozíků tak, aby tvořily celek s
  nejkratší cestou po skladu při vytížení všech lidí za splnění všech
  pravidel. Dále položky na vozíku seřadit do seznamu rovněž do co
  nejkratší trasy.
  ```

- **Statistiky**
    - Řeším prostor o cca 30000 adresách a 20000 produktech.
    - Do algoritmu mi může přijít 1 položka i 2000 položek - pro vše se chovám stejně
    - Každému vozíku se přiřazovalo 8 HD
    - Nejvíce bývá aktivních 40 lidí (=40 vozíků) v sezóně, mimo sezónu 15 (=15 vozíků).

Base-line řešení

Úvod

Protože aktuální implementace algoritmu, jehož optimalizace je řešena v této práci, téměř neexistuje (vozíky jsou vytvářeny a data v nich řazena čistě náhodně, pouze na bázi splnění kritérií popsaných výše), bylo

nutné vytvořit prvotní řešení, které mělo za cíl stát se referenčním a tak poskytnout porovnání pro výslednou implementaci algoritmu.

Popis obecné části implementace

Tato část byla psána takovým způsobem, aby byla znovupoužitelná pro úpravy algoritmu pro finální řešení.

Její průběh je složen z několika částí. První je zaměřena na samotné načtení dat a vytvoření objektu, skrz který je možno ovlivňovat základním způsobem průběh výpočtu.

Přílohy

- [1]

n / f(n)	log(n)	n	n log(n)	n^2	2^n	n!
10	0.003 µs	0.01 µs	0.033 µs	0.1 µs	1 µs	3.63 ms
20	0.004 µs	0.02 µs	0.086 µs	0.4 µs	1ms	77.1 years
30	0.005 µs	0.03 µs	0.147 µs	0.9 µs	1s	8.4×10^{15} years
40	0.005 µs	0.04 µs	0.213 µs	1.6 µs	18.3 min	
50	0.006 µs	0.05 µs	0.282 µs	2.5 µs	13 days.	
100	0.007 µs	0.1 µs	0.644 µs	10 µs	4×10^{13} years	
1,000	0.010 µs	1 µs	9.966 µs	1ms		
10,000	0.013 µs	10 µs	130 µs	100ms		
100,000	0.017 µs	0.10 ms	1.67 ms	10s		
1,000,000	0.020 µs	1ms	19.93 ms	16.7 min		
10,000,000	0.023 µs	0.01 s	0.23 s	1.16 days		
100,000,000	0.027 µs	0.1 s	2.66 s	115.7 days		
1,000,000,000	0.030 µs	1s	29.90 s	31.7 years		

Zdroje

- [1] https://subscription.packtpub.com/book/application-development/9781785884504/8/ch08lvl1sec54/evaluating-runtime-complexity
- [2] Honzík,M J.: Algoritmy a datové struktury, 2018