Fakulta riadenia a informatiky

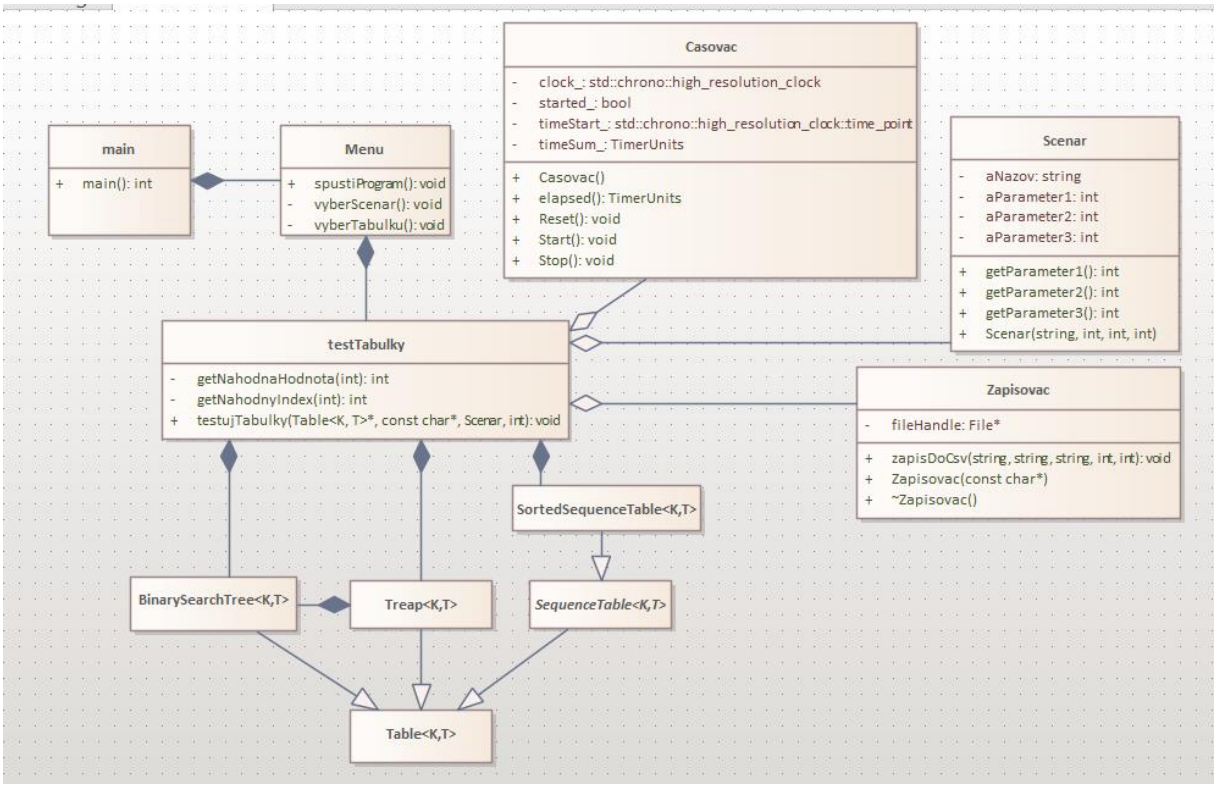
Informatika

Domáce zadanie 3

Analýza výkonu tabuliek

Peter Kulas
5ZYI21

UML diagram:


Casovac
- clock_: std::chrono::high_resolution_clock
- started_: bool
- timeStart_: std::chrono::high_resolution_clock::time_point
- timeSum_: TimerUnits
+ Casovac()
+ elapsed(): TimerUnits
+ Reset(): void
+ Start(): void
+ Stop(): void
main
+ main(): int
Menu
+ spustiProgram(): void
- vyberScenar(): void
- vyberTabulku(): void
Scenar
- aNazov: string
- aParameter1: int
- aParameter2: int
- aParameter3: int
+ getParameter1(): int
+ getParameter2(): int
+ getParameter3(): int
+ Scenar(string, int, int, int)
testTabulky
- getNahodnaHodnota(int): int
- getNahodnyIndex(int): int
+ testujTabulky(Table<K, T>*, const char*, Scenar, int): void
Zapisovac
- fileHandle: File*
+ zapisDoCsv(string, string, string, int, int): void
+ Zapisovac(const char*)
+ ~Zapisovac()
SortedSequenceTable<K,T>
BinarySearchTree<K,T>
Treap<K,T>
SequenceTable<K,T>
Table<K,T>

Popis testovania tabuliek:

Testovacia trieda sa skladá z 3 metód 2 metódy sú určené na generovať čísiel, pričom jedna generuje od 1 po zadané x a druhú využívam na generovanie indexov od 0 po zadané x. Ako tretia metóda je samotná hlavná metóda testujTabuly, ktorá berie ako parametre (Table<T>* paTabulka, const char* nazovSuboru, Scenar Scenare[], int paSize). Na začiatku si pomocou dynamic_castu zistím o aký typ tabulky sa jedná, či ide o SortedSequenceTable, BinarySearchTree alebo Treap. Typ tabulky si zisťujem len kvôli zápisu, pre prehľadnosť. Keďže máme vykonať operácie v nejakom určenom pomere budem náhodne generovať čísla od 1 po 100 a ak náhodná hodnota „spadá“ do daného pomeru vykonám danú operáciu. Napríklad v scenári A máme daný pomer operácií 30:60:10. Ak náhodne vygenerovaná hodnota bude z intervalu <0,30> tak sa vykoná operácia Vlož na základe kľúča. Pri spracovaní metódy vlož si kontrolujem, či sa už kľuč v tabuľke nachádza, ak nie vykoná sa operácia instert ak nie pokračujeme v cykle. Na začiatku si vždy vygenerujem náhodný kľúč z intervalu <1,100000> a pri vkladaní do tabuľky ho vložim do vectora kľúčov, odiaľ ho potom náhodne vytiahnem pri operáciach sprístupni na základe kľúča a odstráň na základe kľúča. Ak je náhodná hodnota v intervale (30,90) tak sa vykoná operácia sprístupni na základe kľúča, ktorá sa opäť vykoná len vtedy ak daný kľúč sa nachádza v tabuľke. Ak sa náhodná hodnota nachádza v intervale <90,100> vykoná sa operácia delete. Pričom zmaže prvok z tabuľky. Pri každej operácií kontrolujem či sa nachádza náhodný kľúč v tabuľke, ak sa pri operáciach vymaž a spristupni data daný kľúč v tabuľke nenachádza, operácia sa preskočí a inkrementuje sa premenné preskoceneOperacie. V prípade insertu ak sa už daný kľúč v tabuľke nachádza tak opäť pokračujem v cykle a inkrementuje sa premenná preskoceneOperacie. Taktiež preskakujem operáciu a inkrementujem premennú ak pri operáciach vymaž a spristupni data je tabuľka prázdna. Tým pádom sa celý cyklus vykoná 100 000 + preskoceneOperacie krát. Na výber jednotlivých scenárov som vytvoril triedu Scenár, ktorá ma 3 int atribúty a 1 string pre názov scenára. Vytvorím 3 scenáre, pričom každý má požadované hodnoty a uložím ich do poľa. Celý vyššie opísaný for-cyklus, je obalený ešte 1 for-cyklom, ktorý sa vykoná x krát, podľa počtu scenárov.

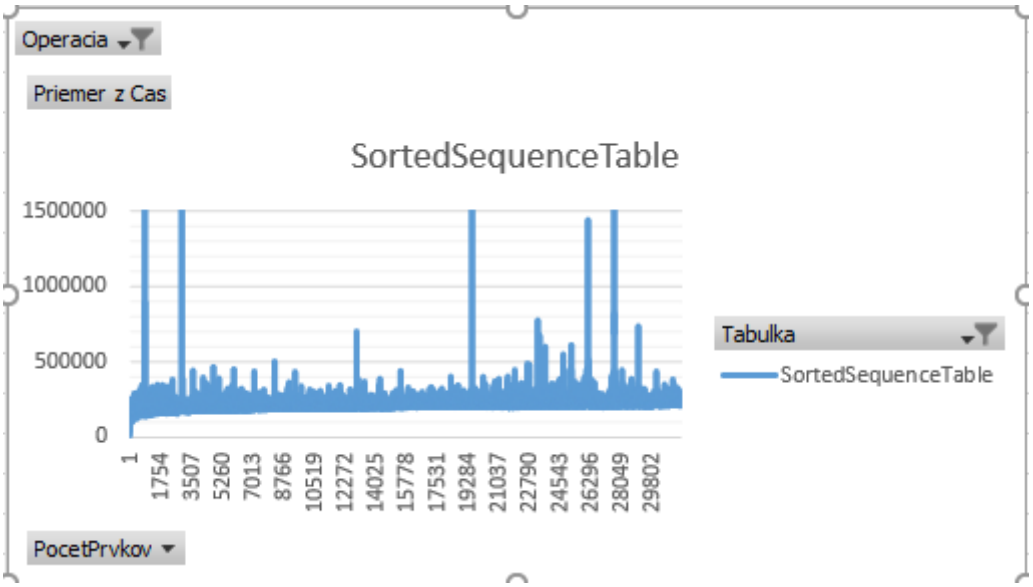
Tab. 2 Testovacie scenáre pre ADT tabuľka

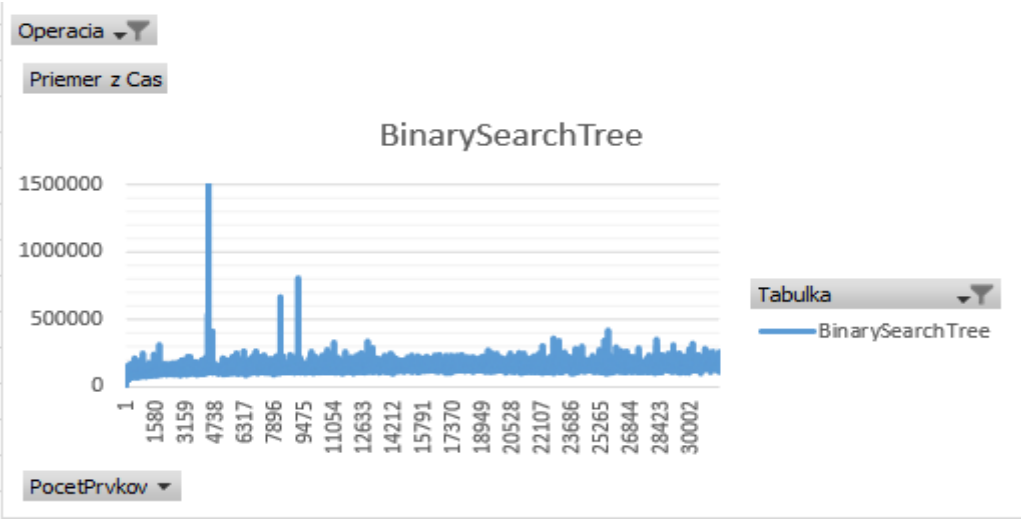
Scenár	**Podiel operácií**		
	Vlož na základe kľúča	*Sprístupni na základe kľúča*	*Odstráň na základe kľúča*
A	30	60	10
B	45	10	45

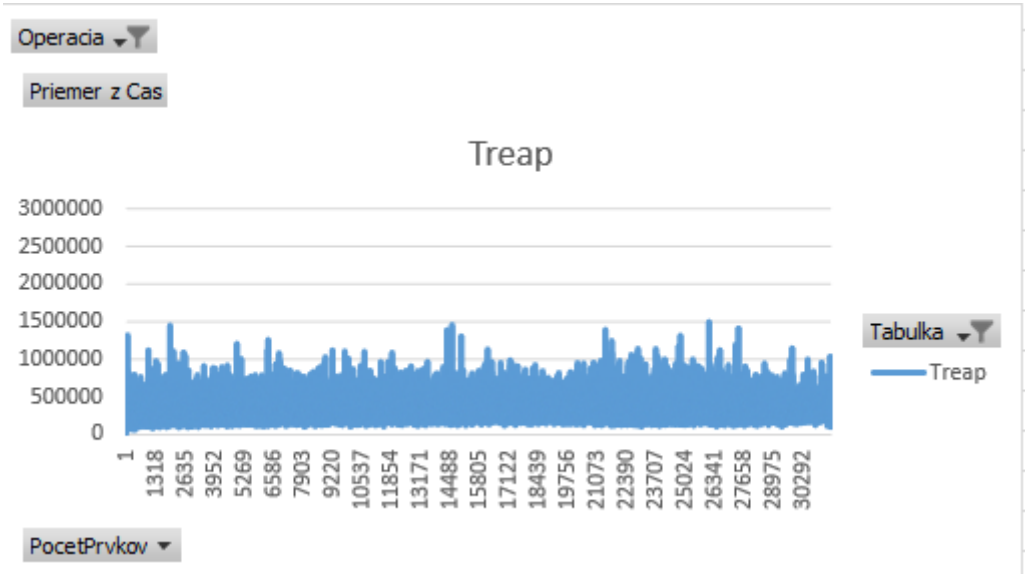
Zaznamenávanie údajov do CSV súboru:

Ako bolo napísané v zadaní výsledky vykonávania jednotlivých operácií, resp. časy v nanosekundách zapíšem do .csv súboru, ktorý neskôr spracujem v tabuľkovom programe Excel. Údaje budem do .csv súboru zapisovať vo formáte [Scenar], [TypStruktury], [Operacia], [PocetPrvkov],[Cas[ns]]. Na zápis som si vytvoril triedu Zapisovac, ktorá berie 5 parametrov, 3 stringy a 2 int. Všetky scenáre sú zapísane pod sebou v jednom súbore, takže .csv súbor bude obsahovať 900 000 riadkov, to znamená 3 vykonané scenáre pre každý z 3 tabuliek (BinarySearchTree, SortedSequenceTable, Treap) . Grafy vypracujem na základe kontingenčnej tabuľky, pričom tu vytvoríme nasledovne: Ak nemáme hlavičku tak si nad dáta zapíšeme názvy jednotlivých stĺpcov. Následne si označíme všetky stĺpce okrem typ scenára a klikneme na vložiť na kontingenčnú tabuľku. V pravom panely s názvom Polia kontingenčnej tabuľky, označíme všetky stĺpce kliknutím na checkbox. Následne tieto stĺpce rozdelíme do polí. Ja som dal do filtrov operáciu, Legenda obsahuje typ štruktúr. Os obsahuje počet prvkov a v hodnotách sa nachádza čas. Na čas klikneme ľavým tlačidlom myši, klikneme na nastavenie poľa hodnoty a vyberieme možnosť priemer. Nakoniec klikneme na vytvoriť kontingenčného grafu v možnosti analýza kontingenčného grafu.

1. **Operácia vkladanie prvku na základe kľúča**

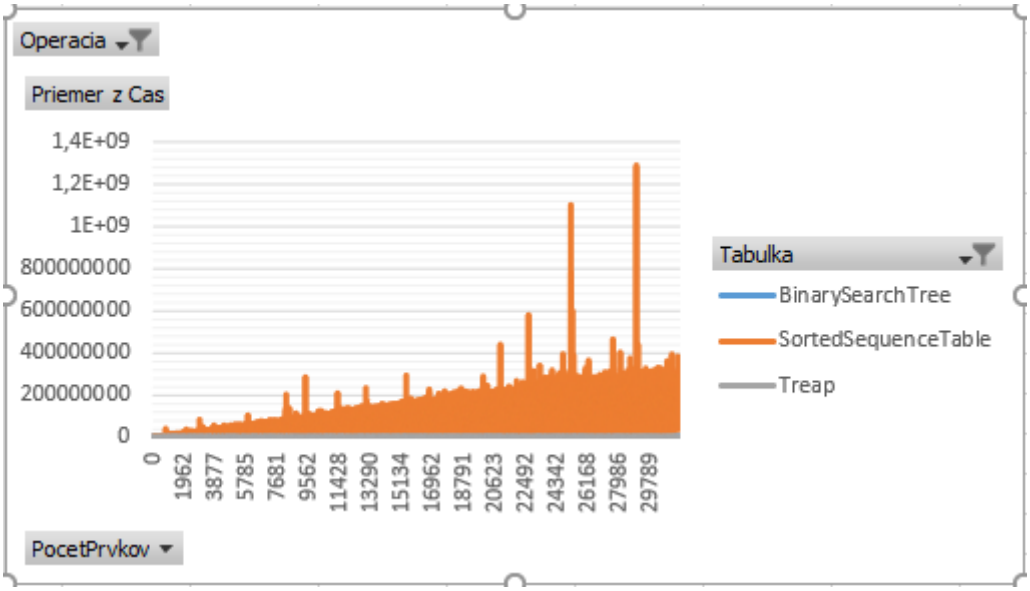


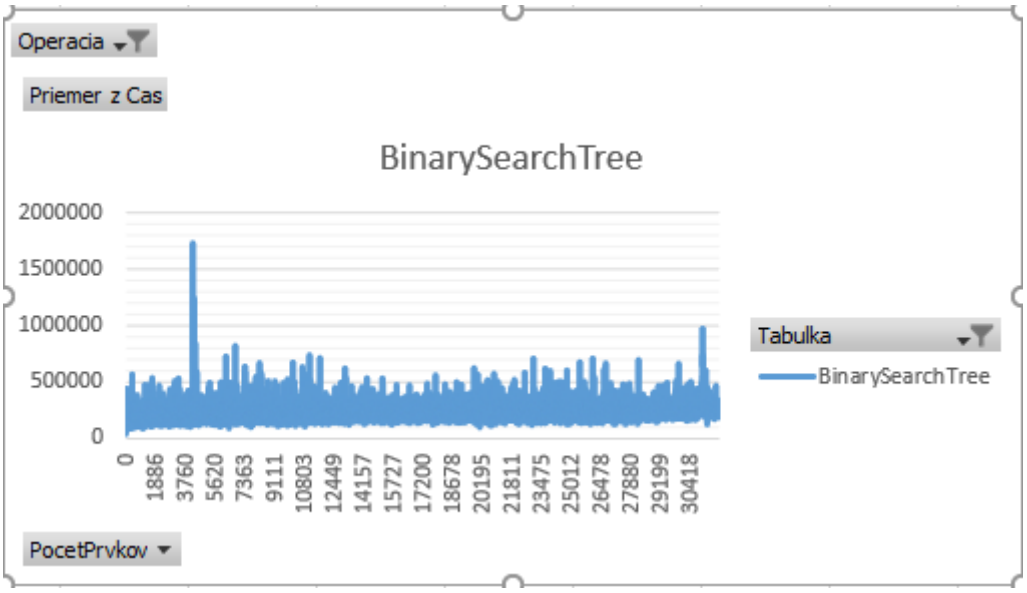


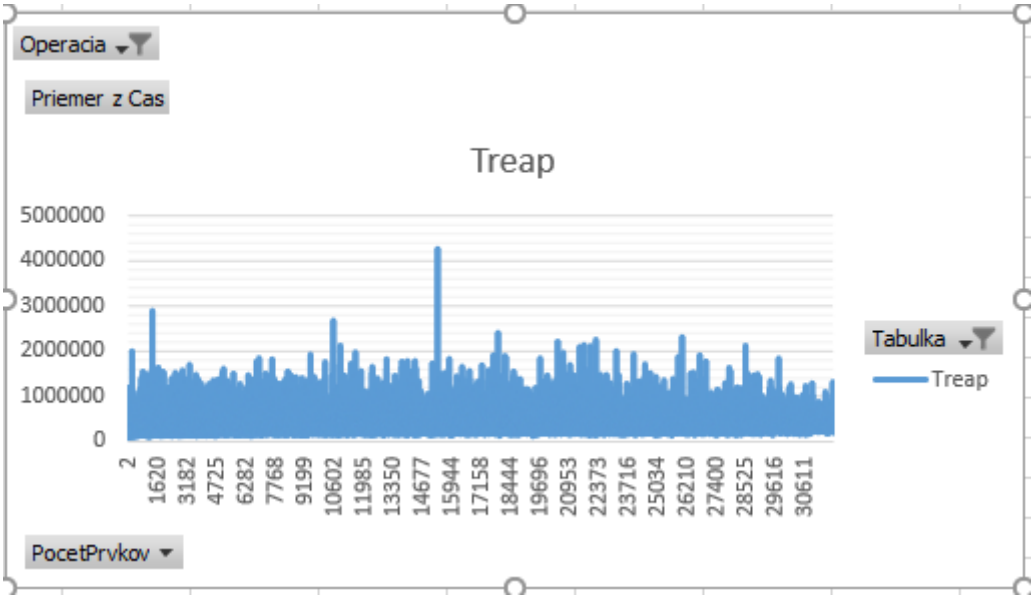


Ako môžeme vidieť z grafov utriedená sekvenčná tabuľka a binárny vyhľadávací strom sú rýchlejšie ako Binárny strom s haldovým usporiadaním, ďalej len trep a ich zložitosti odhadujem na O(N), keďže vidno mierne stúpanie. Čo sa mi ale vôbec nepozdáva je graf treap-u, ktorý by mal byť najrýchlejší z týchto 3 tabuliek a jeho zložitosť by mala byť pri O(log n). A myslím si, že by mal byť najrýchlejší avšak momentálne v porovnaní s ostatnými je najpomalší, niekde nastala chyba avšak kvôli nedostatku času som ju neriešil.

2. **Operácia mazanie prvku na základe kľúča**

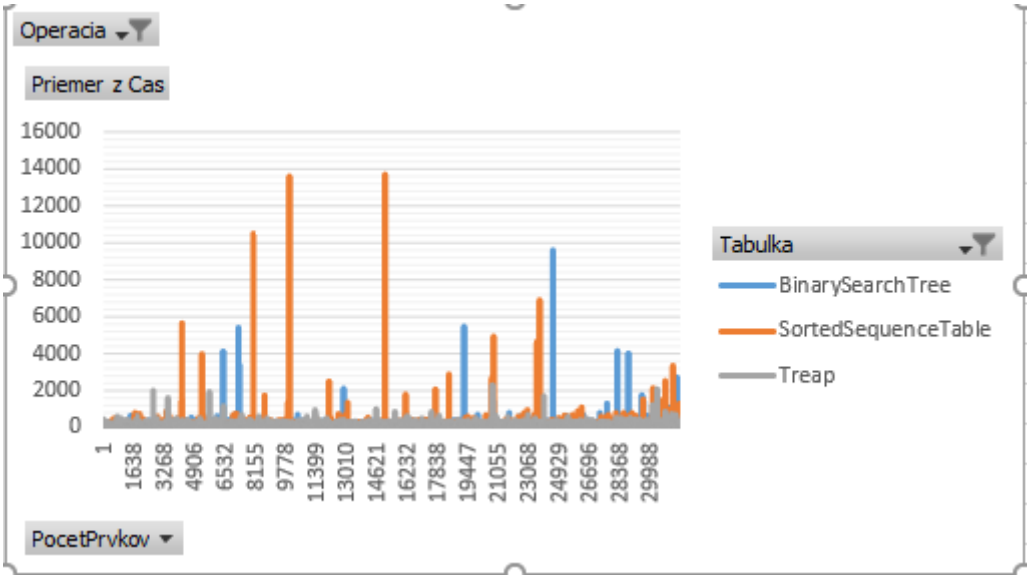






Pri mazaní prvkov na základe kľúča vidíme, že utriedená sekvenčná tabuľa je najpomalšia, kde nastáva zložitosť O(N), ktorú som aj očakával. Z ďalších 2 grafov vidíme ako majú Binárny vyhľadávací strom a Treap časovú zložitosť O(log N), čo vlastné sedí s teoretickým predpokladom.

3. Operácia sprístupni dáta na základe kľúča



Z grafu pre operáciu sprístupnenia dát na základe kľúča som vydedukoval že všetky 3 štruktúry, Binárny vyhľadávací strom, Usporiadaná sekvenčná tabuľka a aj Treap malú zložitosť O(log N), pričom Treap vyzerá byť jasne najrýchlejší. Výkyvy pri Binárnom vyhľadávacom strome a usporiadanej sekvenčnej tabuľke sú pravdepodobné spôsobené nejakými vonkajšími vplyvmi.