AVL strom

David Chudožilov

Vycházím z na cvičení už naprogramovaného binárního vyhledávacího stromu

- Funkce a metody: `avl_tree()`, `empty()`, `remove_all(node*)`, `insert(const T&)`, `write()`, `remove_node(const T&)`, `treeprint()`, `find(const T&, node*& parent)`

AVL strom

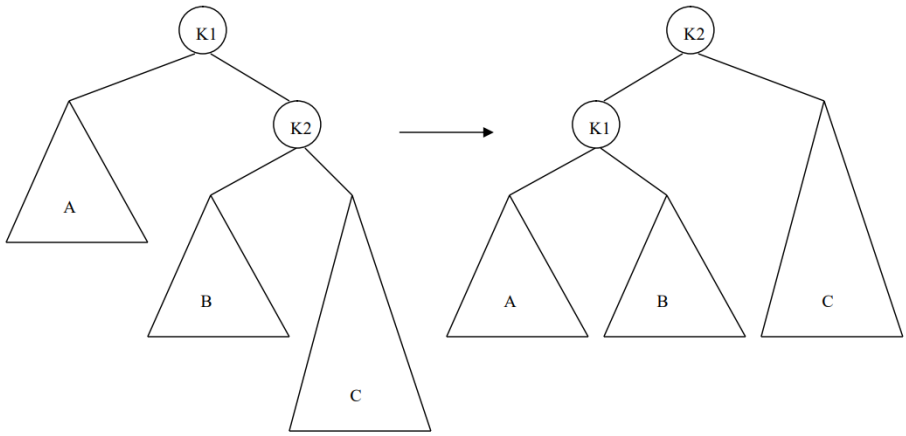
- Modifikace binárního vyhledávácího stromu, která udržuje tzv. balanční faktor každého vrcholu velký maximálně 1
- Díky tomu je možné udržet časovou náročnost vyhledávání hodnoty ve stromě O(log n) ve všech případech, tedy nemůže nastat situace, kdy strom bude silně nevyvážený a začne se přibližovat spojovému seznamu, který má složitost vyhledáváni O(n)

Balanční faktor

- `int balance` - prvek třídy `avl_tree::node`
- Definován jako hloubka levého podstromu minus hloubka pravého podstromu
- Pokud jeho hodnota v absolutní hodnotě pro libovolný vrchol přesáhne 1, strom již není vybalancovaný a je třeba ho znovu vyvážit pomocí rotací

Levá rotace

- Odpovídající metoda `left_rotation(node* node_, node* parent)`
- Rotace se provádí na vrcholu `node_`, tedy tento vrchol je ten, který se po rotaci bude nacházet o level níž
- Argument `node* parent` vždy značí rodiče vrcholu `node_`, případně `nullptr` pokud `node_` je kořen stromu

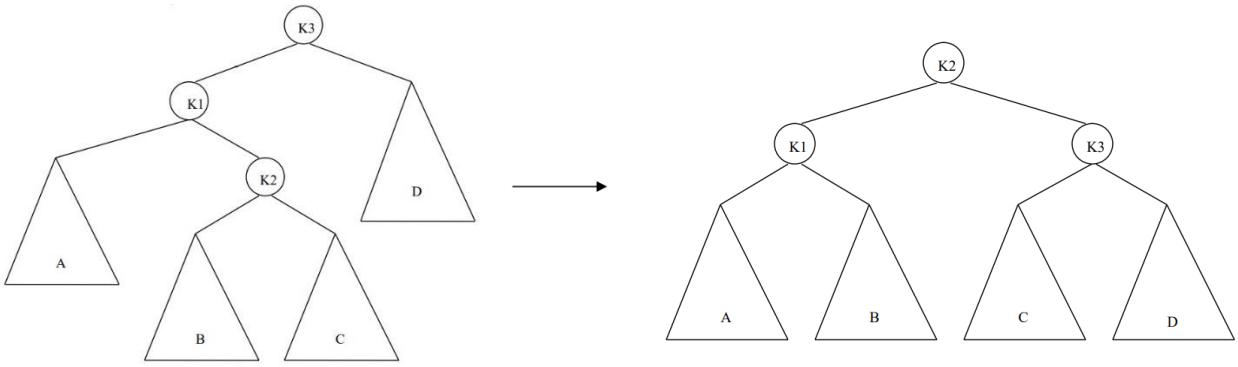


Pravá rotace

- Odpovídající metoda `right_rotation(node* node_, node* parent)`
- Analogické provedení jako levá rotace

Levo-pravá rotace

- Metoda `left_right_rotation(node* node_, node* parent)`



- Na levém potomku vrcholu node_ nejprve provedena levá rotace, na vrcholu node_ následně pravá rotace

Pravo-levá rotace

- Metoda `right_left_rotation(node* node_, node* parent)`
- Na pravém potomku vrcholu `node_` nejprve provedena pravá rotace, na vrcholu `node_` následně levá rotace

Obecný algoritmus vyvažování

a) Pokud je nevybalancovaný kořen stromu
- Balanční faktor < -1 – provede se na kořeni levá rotace
- Balanční faktor > 1 – provede se na kořeni pravá rotace

b) Pokud je nevybalancovaný levý potomek daného vrcholu
- Balanční faktor < -1 – na daném vrcholu se provede levo-pravá rotace
- Balanční faktor > 1 – na daném vrcholu se provede pravá rotace

c) Pokud je nevybalancovaný pravý potomek daného vrcholu
- Balanční faktor < -1 – na daném vrcholu se provede pravá rotace
- Balanční faktor > 1 – na daném vrcholu se provede pravo-levá rotace

Implementace vyvažování nevybalancovaného stromu

- Vybalancovanost se kontroluje a případně i provádí vyvažování po každém vložení nebo smazání vrcholu
- Funkce `balance(node* end_node)`
- `node* end_node` je právě přidávaný vrchol, pokud prováděná operace je vkládání vrcholu do stromu, nebo rodič právě mazaného vrcholu, pokud se provádí mazání
- Funkce prochází strom od kořene po `end_node` a kontroluje balanční faktor
- Jakmile objeví nevybalancovaný vrchol, provede příslušnou rotaci a přepočítá nové balanční faktory pomocí funkce `recalculate_after_balance(node* node_)`, kde argument node ukazuje na stejné místo ve struktuře stromu, kde se před rotací nacházel vrchol, na kterém byla provedena rotace
- Tato funkce rekurzivně přepočítá balanční faktory `balance` a hloubku `depth` celého podstromu vycházejícímu z vrcholu, který se nyní nachází na místě, kde se předtím nacházel vrchol, na kterém byla provedena rotace, a dále přepočítá `number_of_levels` celého stromu