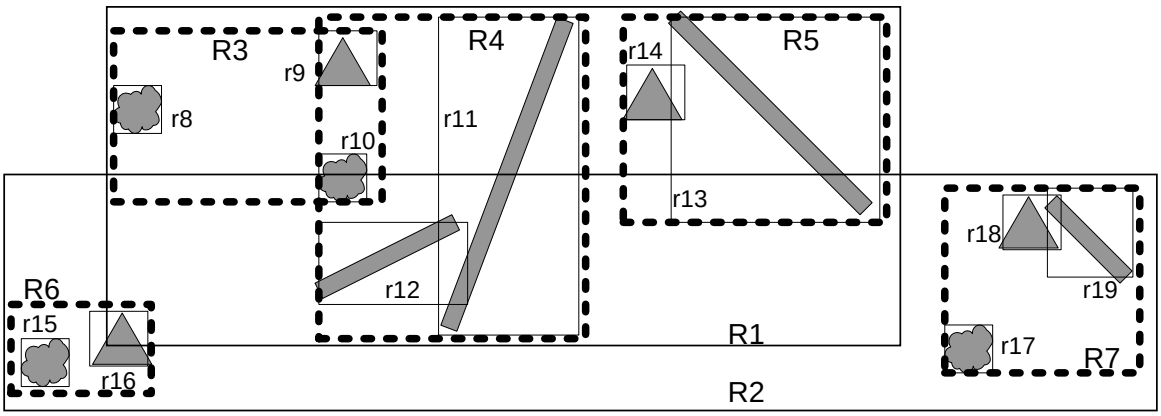
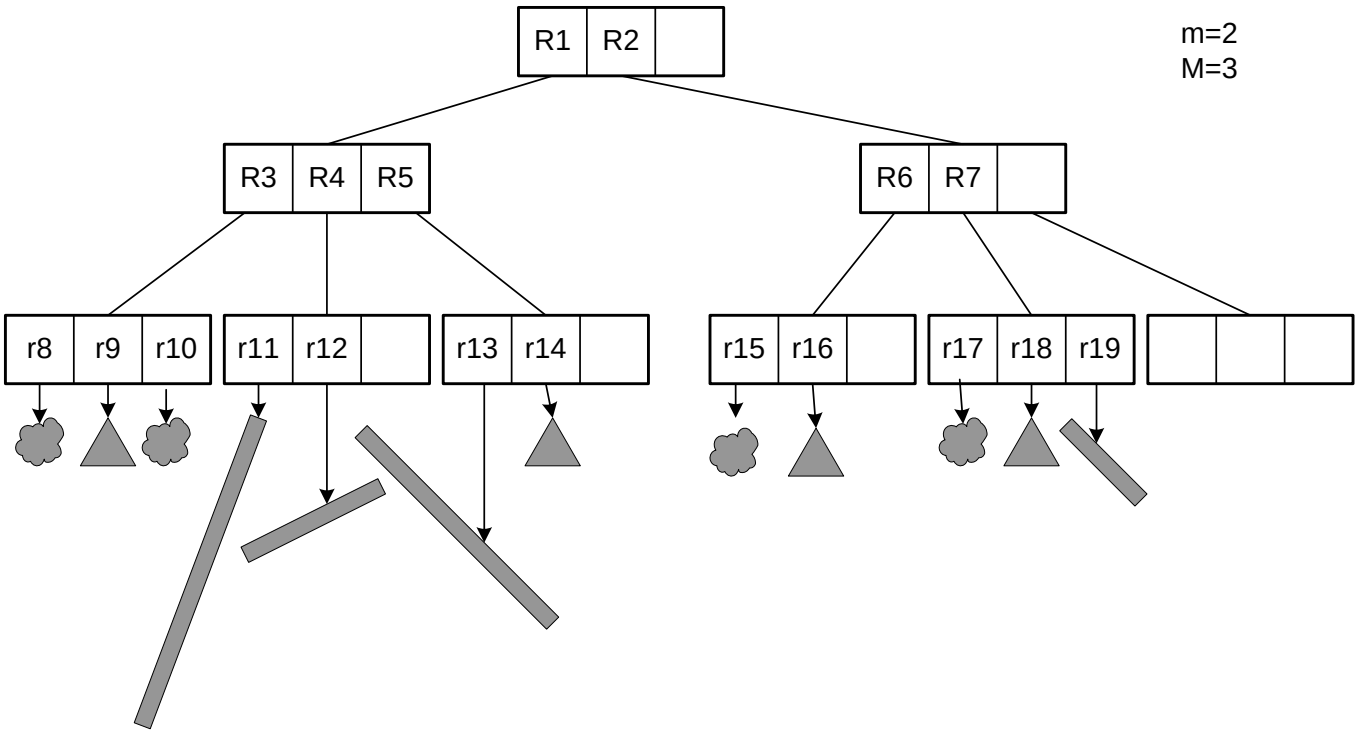
R- strom

R ..... rectangle (pravouholník)

Principiálna vlastnosť: hierarchické zhlukovanie do blokov na základe blízkosti.
Podporuje viacrozmerné intervalové vyhľadávanie.
Štruktúra: podobná B - stromu (B^+ - stromu)
Využitie: najmä (nielen) v „priestorových“ (spatial) aplikáciách
Prudký vývoj - najmä zníženie (najhoršej) zložitosti
***„Najmenší ohraničujúci pravouholník“* (NOP) (minimal bounding rectangle) R1...R7, r8...r19 nahrádza skupiny objektov respektíve objekty.**





Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

NOP sú viacrozmerné intervaly, teda ide o intervalové viacrozmerné vyhľadávanie.
Objekty v databáze (uložené na disku – napr. neutriedený súbor) sú určené svojim identifikátorom (adresou) a svojim NOP.

Odkazy (adresy) na všetky objekty sú v listoch R - stromu a majú tvar: (I, jedinečný identifikátor objektu (resp. adresa na disk, kde sa objekt nachádza)), kde I je NOP objektu.
Vnútorné uzly R - stromu majú tvar (I, adresa syna na disku), kde I pokrýva NOPy všetkých objektov v podstrome „syn“.

Vlastnosti:

- **Výškovo vyvážený strom podobný B – stromu.**
- **Samoorganizujúca sa štruktúra.**
- **Vrcholy stromu sú bloky na disku – teda nie v operačnej pamäti ako napr. quad strom a k-d strom.**
- **Na rozdiel od B - stromu nie sú záznamy v bloku usporiadané.**
- **Počet záznamov v uzle je od m po M (okrem koreňa), kde:**
  - **M – je zvolený tak, aby sa veľkosť bloku rovnala veľkosti clusteru,**
  - **m ≤ M/2 – zvolí sa v stanovenom rozsahu.**
- **Výška R - stromu je $\log_m N - 1$ kde N je počet objektov.**

Štruktúra je primárne určená na implementáciu na disk, ale vďaka skvelým vlastnostiam a jej flexibilite je často implementovaná aj ako štruktúra nachádzajúca sa v operačnej pamäti!

Dôležité je uvedomiť si, že štruktúra môže byť n-rozmerná (pre názornosť je vysvetlená na dvojrozmernom príklade).

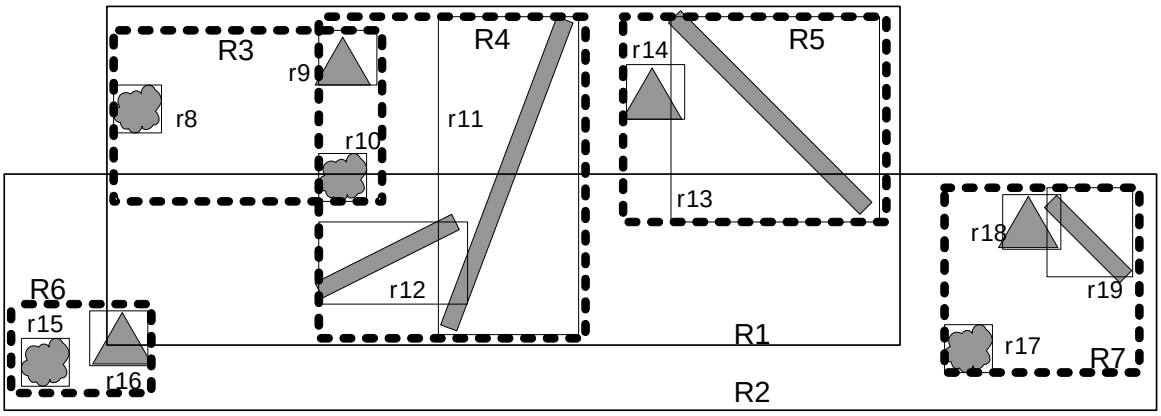
<u>Find (vyhľadanie):</u>

Majme R - strom s koreňom *T,* nájdi všetky objekty, ktorých NOP je obsiahnutý vo vyhľadávacom NOPe *S*.

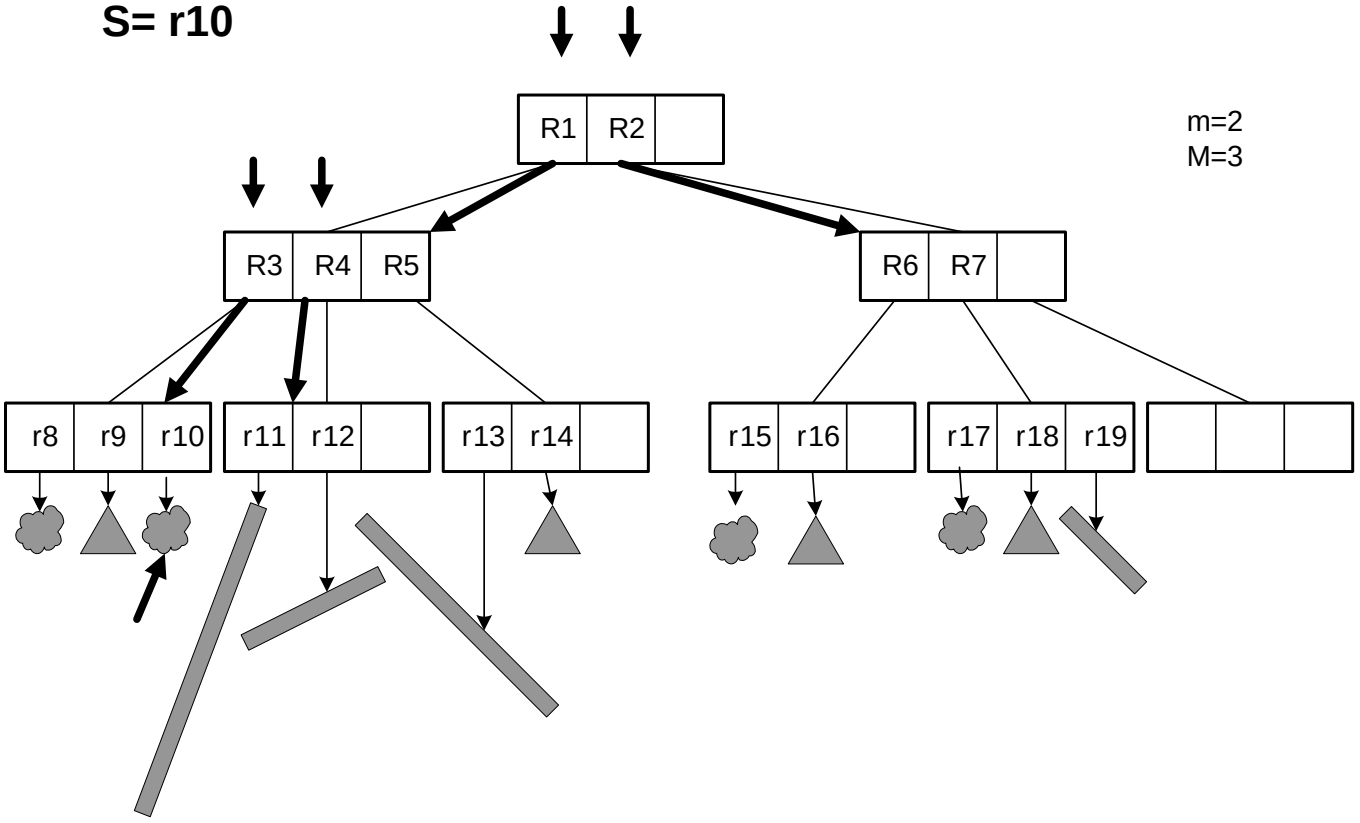
- **Ak *T* nie je listom, nájdi v *T* všetky záznamy, ktorých NOP sa prekrýva s *S* (majú neprázdny prienik).**
- **Pre každý prekrývajúci sa záznam urob vyhľadanie v jeho synovi (podstrome).**
- **Ak *T* je list, prehliadni všetky jeho záznamy a vráť ako výsledok tie, ktoré sú obsiahnuté v *S*.**

Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

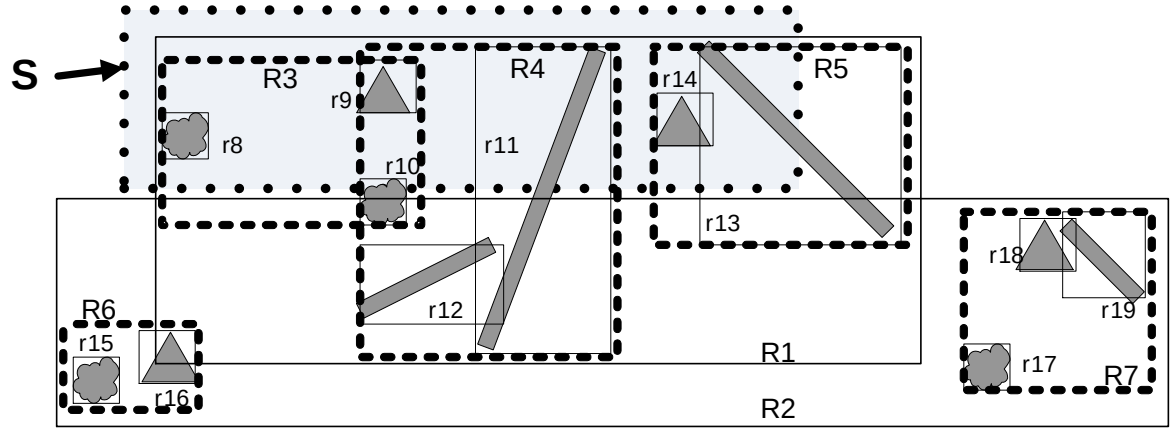
Hľadanie je v princípe totožné s hľadaním v B-strome, avšak tu môže byť potrebné prehliadnuť viacero podstromov. To spôsobuje, že nie je možné garantovať najhoršiu zložitosť ako v B-strome.

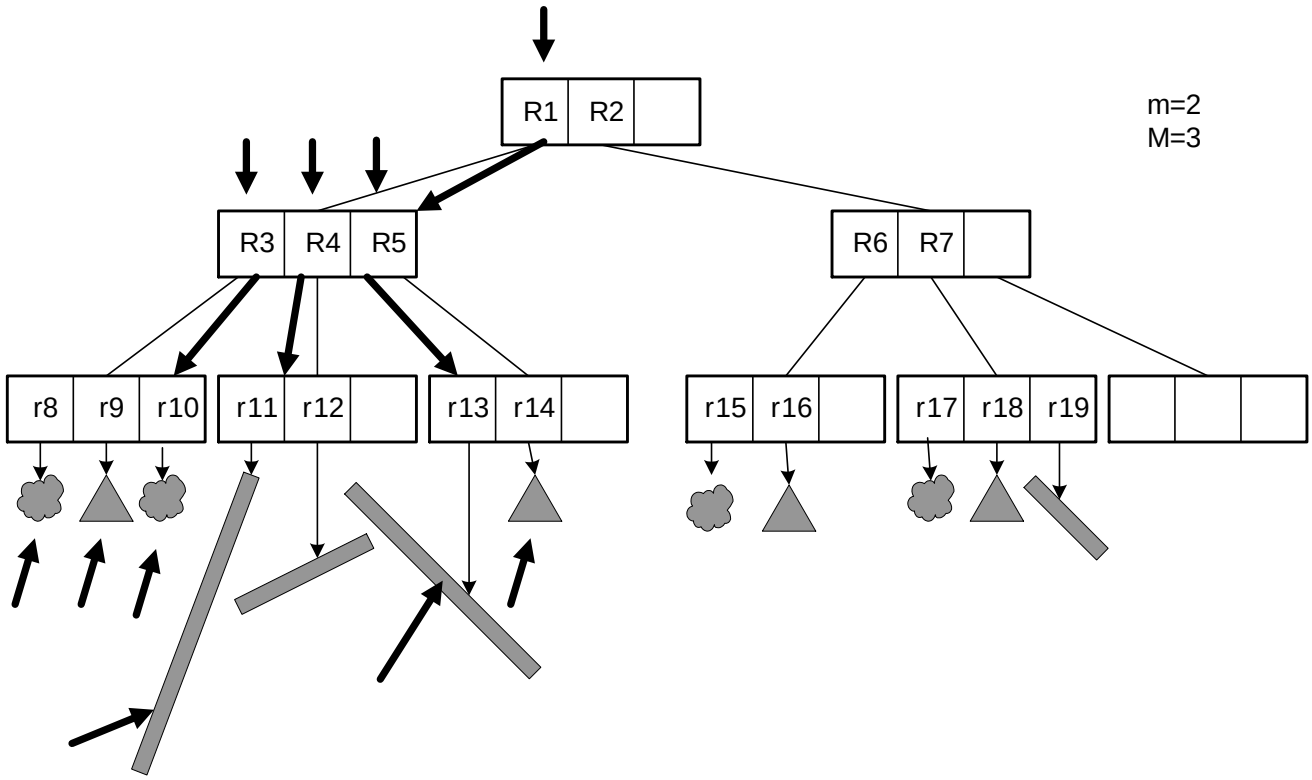


S= r10



Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.





Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

Insert (vkladanie):

Vložíme záznam o objekte E (I, p), kde I je jeho NOP a p je adresa odkazujúca na dáta umiestnené v inom súbore na disku.

- **Nájdi pozíciu – list T pre vloženie:**
  - **Nastav lokálnu premennú T na koreň. Ak T je list, vráť T.**
  - **Vyber záznam F v T, ktorý si po pridaní E vyžiada najmenšie zväčšenie jeho NOP.**
  - **Postup opakuj na T = syn (F), atď. až na list.**
- **Ak teraz list T má dosť miesta pre E, vlož E. Inak T rozdeľ (split) a postupuj smerom nahor až po prípadné rozdelenie koreňa, technicky presne podľa pravidiel B - stromu. Na rozdiel od B - stromu je tu pre efektívnosť rozhodujúce, ktoré objekty vybrať do každej z dvoch častí po rozdelení, teda ako realizovať split.**

Delete (mazanie):

Podobne ako v B - strome. Po odstránení záznamu musíme R strom upraviť, na to sa používa algoritmus :

a. **Nastavíme N=L, kde L je list, z ktorého sme vymazali záznam. Nech Q je množina vyradených vrcholov(zatiaľ je prázdna).**
b. **Ak vrchol N má menej záznamov ako m, odstránime oblasť, ktorá ho pokrývala z otca tohto vrcholu a pridáme vrchol N do množiny Q, inak choď na krok e).**
c. **Upravíme NOP v predkovi P vrcholu N po odstránení oblasti, ktorá ho pokrývala. Zmenšíme ju na najmenšiu možnú.**
d. **Nastav N=P a opakuj od kroku b)**
e. **Znovu vložíme všetky záznamy vrcholov z množiny Q (Záznamy z vyradených listov sú znovu vložené do stromu).**

Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

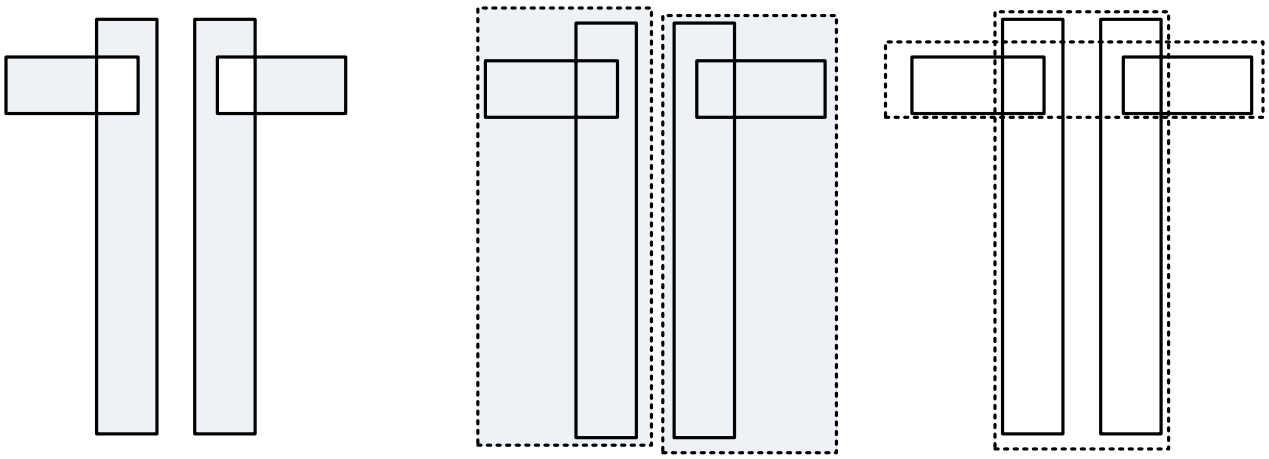
Operácia Split:

Ak pri vkladaní nie je miesto pre vloženie nového záznamu je nutné realizovať split, ktorá rozdelí záznamy na dve skupiny, čím sa umožní vytvorenie nového vrcholu.

Efektívnosť hľadania je tým lepšia:

- **čím je menší súčet plôch NOPov**
- **čím je menšia „nevyužitá“ plocha (súčet plôch NOPov mínus súčet plôch objektov)**
- **čím menej NOPov sa prekrýva**

Cieľ: optimálny alebo suboptimálny split



zle **dobre**

Pri R-strome existujú 3 základné algoritmy operácie *split*:

1. **Optimálny split - hrubá sila (preskúmanie všetkých splitov) – zložitosť splitu je $O(2^{M-1})$. Keďže M je často väčšie číslo je využitie tejto techniky ťažko možné. Z tohto dôvodu sa pri operácií split využívajú heuristiky.**

2. **Kvadratická heuristika – Z prvkov vrcholu (je ich M+1) sa vyberú dva, ktoré by pri zaradení do jediného NOPu pokrývali najväčšiu oblasť. Vyberieme teda dva prvky, ktoré sa najmenej hodia na umiestnenie do jedinej skupiny. Následne tvoria tieto prvky základy dvoch skupín. Ostatné prvky priradíme do jednej so skupín, kde najmenej zväčšia NOP.**

Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

Vyberáme ich v poradí podľa rozdielu plochy zväčšenej oblasti.

3. **Lineárna heuristika – Pre každú dimenziu vyber NOP, ktorý má najväčšiu kratšiu stranu a prvok, ktorý má najmenšiu dlhšiu stranu. Z takto vytvorených dvojíc vyber tú, ktorá priestor k hľadiska príslušnej dimenzie rozdeľuje najrovnomernejšie. Ostatné prvky priradíme do jednej so skupín, kde najmenej zväčšia NOP.**

Heuristiky - problém s najhoršou zložitosťou.

V prvej publikovanej verzií (Guttman, A.,1984) boli navrhnuté tri vyššie popísané spôsoby realizácie splitu. Samotná štruktúra je dodnes veľmi často využívaná a jej zdokonalenia sú zamerané najmä na čo najefektívnejšiu realizáciu operácie split. Experimentovalo sa aj s inými ohraničujúcimi objektmi (namiesto NOPov použiť napr. kruhy – M-strom), ale aj s „naklápaním“ rovín NOPov podľa prevažujúcej orientácie objektov.

Štruktúra bola vysvetlená na dvojrozmernom priestore, ale je samozrejme použiteľná aj pre 3D aplikácie.

Packed R strom

Pôvodný R strom je navrhnutý pre dynamické operácie vkladania a odoberania. Ak vieme vopred aké dáta ideme vkladať môžeme dosiahnuť lepšie využitie pamäte (priestoru na disku). Zlepšenie môžeme dosiahnuť použitím “packing” algoritmov. Cieľom týchto algoritmov je väčšie zaplnenie vrcholov, čo spôsobí zníženie výšky stromu. Základom je vhodne utriediť záznamy. Mnohé algoritmy sú založené na krivkách vyplňujúcich priestor, napríklad Hilbertová krivka, alebo Z krivka.

Základný algoritmus

Záznamy sú utriedené rastúcom (príp. klesajúco) podľa x-ovej alebo y-psilonovej hodnoty dolného ľavého, horného pravého rohu

Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

NOP záznamu alebo podľa stredu NOPu. Následne sa na základe utriedených dát vytvárajú listy s maximálnou kapacitou. Po vložení všetkých záznamov do listov, prejdeme o úroveň vyššie a z listov postupne vytvárame vnútorné uzly s maximálnou kapacitou .
A takto postupujeme až, kým nám neostane len jeden uzol, ktorý prehlásime za koreň.

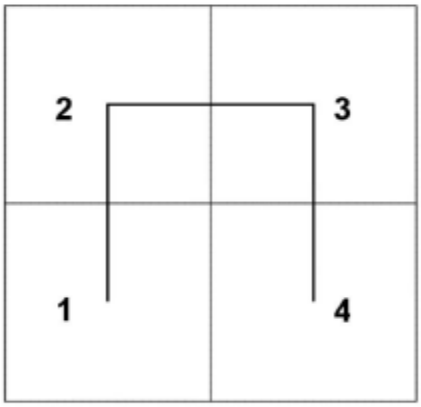
Algoritmus najbližšieho suseda

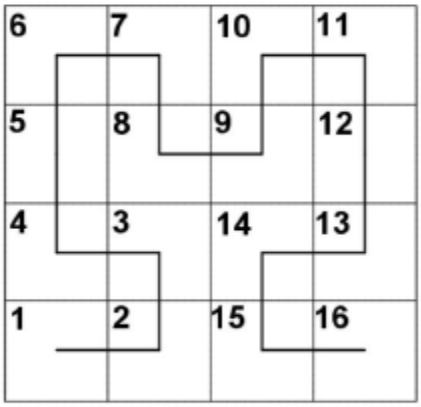
Záznamy utriedime podľa x-ovej súradnice. Opäť môžeme vybrať hodnotu buď dolného ľavého, horného pravého alebo podľa stredu NOPu. Vyberieme prvý záznam z utriedenej postupnosti a hľadáme najbližších susedov záznamu, kým nezaplníme list. Opakujeme, kým nie sú všetky záznamy vložené do listov. Potom prejdeme o úroveň vyššie, kde podobným spôsobom vytvárame vnútorné uzly s maximálnou kapacitou. Takto postupujeme o úroveň vyššie až po kým nám neostane len jeden uzol, ktorý sa stane koreňom.

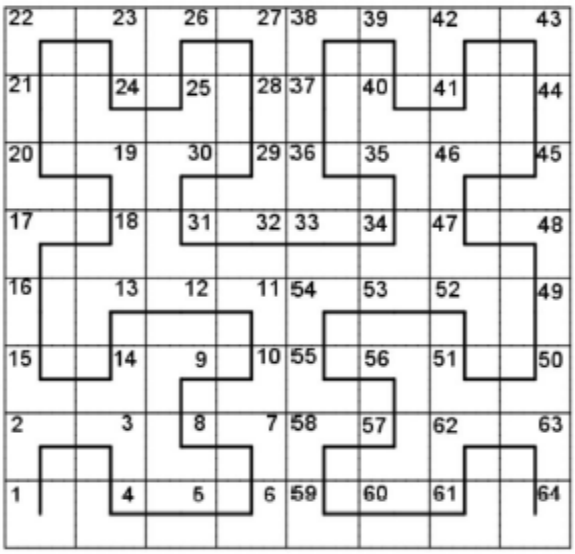
Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

Algoritmus využívajúci Hilbertovu krivku

Tento algoritmus utrieďuje záznamy použitím Hilbertovej krivky. Hilbertová krivka je fraktalová plochu vyplňovacia krivka, ktorá udáva lineárne poradie prechodu viacrozmerným priestorom. Príklad Hilbertovej krivky môžeme vidieť na obrázkoch.







Pre každý záznam je vypočítaná hilbertová hodnota – vzdialenosť od hilbertovej krivky. Známe sú viaceré druhy výpočtu hilbertovej hodnoty, ale najpoužívanejšia je vzdialenosť stredu NOPu od krivky. Po vypočítaní hodnoty, záznamy utriedime rastúco podľa hilbertovej hodnoty. Postupne vytvárame listy pridávaním záznamov z tejto postupnosti, kým nie sú tieto zaplnené. Potom postúpime o úroveň vyššie a vytvoríme vnútorné

Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.

vrcholy maximálnej kapacity z týchto listov. Postupujeme o úroveň vyššie pokým nemáme len jeden uzol, ktorý sa stane koreňom.

Ďalšie modifikácie

1. **R*- tree:**
   Štruktúra vychádza z R-stromu. Pri vkladaní sa v operácií split snažíme minimalizovať celkovú plochu výsledného NOPu, ale aj minimalizovať plochu prekrytia NOPov. V prípade, že je vrchol plný dôjde ešte pred operáciou split k vybratiu všetkých doterajších NOPov a ich opätovnému vloženie do štruktúry (reinsert). Poradie ich opätovného vkladania závisí od od vzdialenosti stredu ich NOPu od stredu NOPu vrchola v ktorom sa práve nachádzajú. Toto môže vyvolať kaskádu opätovných vkladaní. Na každej úrovni sa takýto reinsert robí maximálne raz. Vkladanie má zložitosť O(M*log M)

2. **R+ strom:**
   - **neplatí podmienka minimálneho zaplnenia vrcholu na m,**
   - **NOPy vnútorných vrcholov sa neprekrývajú,**
   - **adresa na objekt môže byť uložená vo viac ako jedinom liste.**
3. **PR - tree (priority R - tree)**
4. **mnoho ďalších**

R stromy sú používané najmä v priestorových aplikáciách (počítačová grafika, geografické informačné systémy, navigačné systémy), ale aj v databázových systémoch.

Materiál slúži výlučne pre študentov FRI ŽU, nie je dovolené ho upravovať, prípadne ďalej šíriť.