A* nebo take A Star

Vypracoval: Jakub Suchý

Pro predmet: Algoritmy a datove struktury I (TIN060)

Zadani

Implementace algoritmu A*, ktera funguje na libovolnem ohodnocenem grafu v rovine -- pokud jsou vahy jednicky, bude to upravene prohledavani do sirky v linearnim case, pokud budou vahy komplikovane, pouzije to implementaci s k-regularni haldou pro k = max(2, round(m / n)).

Co je to A*?

A* je vyhledavaci algoritmus, ktery hleda optimalni cestu, je odvozen od algoritmu vyhledavani do sirky. A to tak, ze pouziva heuristickou funkci.
V rovine je nejsnazsi heuristickou funkci vzdusna vzdalenost dvou bodu.
Pojmy: bod x vrchol rovinneho grafu, funkce g(x),h(x) a f(x), kde g(x) je vzdalenost od bodu x ke startu, h(x) je heuristicka vzdalenost k cili (v nasem pripade vzdusna vzdalenost) a f(x) = g(x) + h(x).

Algoritmus

- Vytvor prazdnou prioritni frontu vrcholu openset, podle f(x)
- Do openset vloz vrchol start s pocatecni cestou nulove delky
- Dokud neni openset prazdny, opakuj:
    - Z openset vyber a odstran vrchol v s nenizsi hodnotou f(v) a nastav vrcholu v closed.
    - Jeli vrchol v cilovym vrcholem, zrekonstruuj a vrat cestu a skonci.
    - Projdi vsechny sousedni vrcholy x vrcholu v, pokud nejsou closed, spocitej f(x) a uloz je do prioritni fronty openset.
    - Pokud je v openset nejaky vrchol vicekrat, ponech ten s nejmensi hodnotou f(x) a ostatni smaz.
- Je-li prioritni fronta openset prazdna, vypis, ze cesta neexistuje.

Slozitost

Pokud heuristicka funkce h(x) navic splnuje, ze h(x) - h(y) <= d(x,y), kde d(x,y) je skutecna vzdalenost mezi body x a y (tzn plati trojuhelnihova nerovnost), pak mame zaruceno, ze h(x) bude monotonni a tim padem mame zaruceno, ze kazdy vrchol navstivime maximalne jednou.
Pro graf G=(n,m) v algoritmu tedy prochazim vsechny vrcholy v case O(n), ale vyuzivam k-regularni haldu jako prioritni frontu pro vrcholy s operacemi O(log n) a nakonec prochazim vsechny hrany grafu v case O(m). Z toho vypliva, ze celkova casova slozitost pro bude slozitost O(n*log n + m).

Vstup

- Pro implementaci pro mrizku je napsana knihovna “Mriz”, ze ktere staci vyrobit dvourozmerne pole typu “Mriz.Vrchol”, pote nadefinovat prekazky a to zmenou property “type” u kazdeho vrcholu, kde ma prekazka byt. A pote uz jen pouzit metodu Mriz.PathFind().
- Pro implementaci pro rovinne grafy je napsana knihovna “Rovina”, ze ktere staci vyrobit graf zadany vrcholy a jeho sousedy. K tomu slouzi datovy typ “Rovina.Vrchol v” a pri pridani sousedu metoda “v.AddSoused()”. A pote uz jen pouzit metodu Rovina.PathFind().

K implementaci

- Kod je napsany v C#, vyuzivajici knihoven .NET a to konkretne System a System.Collections.Generic, ze ktery cerpam nektere datove struktury.
- Psal jsem do kodu hodne komentaru, tak by melo byt jasne, co se kde dela.

Zaver

Tento algoritmus jsem si vybral, kvuli moji posedlosti z jeho znalosti, kterou jsem ziskal a take kvuli jeho brutalni rychlosti na grafech reprezentovanych mrizkou. Zaroven ho budu hojne vyuzivat v me zapoctove praci pro Programovani II a to konkretne pro podobnou adaptaci hry Heroes of Might and Magic.