Dokumentace: Průniková reprezentace grafů - kotvené L tvary

Program vykresluje průnikovou reprezentaci grafu pomocí ukotvených L tvarů. Kotvené L tvary mají tvar písmene L otočeného vpravo, nebo vlevo, přičemž jejich horní konce jsou všechny v jedné horizontální linii.

Uživatelská část

Instalace programu:

Program byl vytvořen v jazyce C++ ve Visual Studiu 15 jako WinForms projekt. Projekt se sestaví jednoduše ve Visual Studiu (projekt se načte otevřením souboru `P1.sln`). Program využívá C++ Standard Library a z C++ Support Library header <msclr\marshal_cppstd.h>.

Ovládání programu:

Program disponuje jednoduchým uživatelským rozhraním zobrazeném na Obrázku 1. Samotný graf lze nahrát dvěma způsoby. Při nahrávání grafu ze souboru (tlačítko `Load Graph`) je nutné dodržet formát, kdy na každé řádce je jedna hrana grafu jako dvojice čísel oddělených mezerou. Vrcholy se číslují od jedničky. Graf se po nahrání vykreslí.

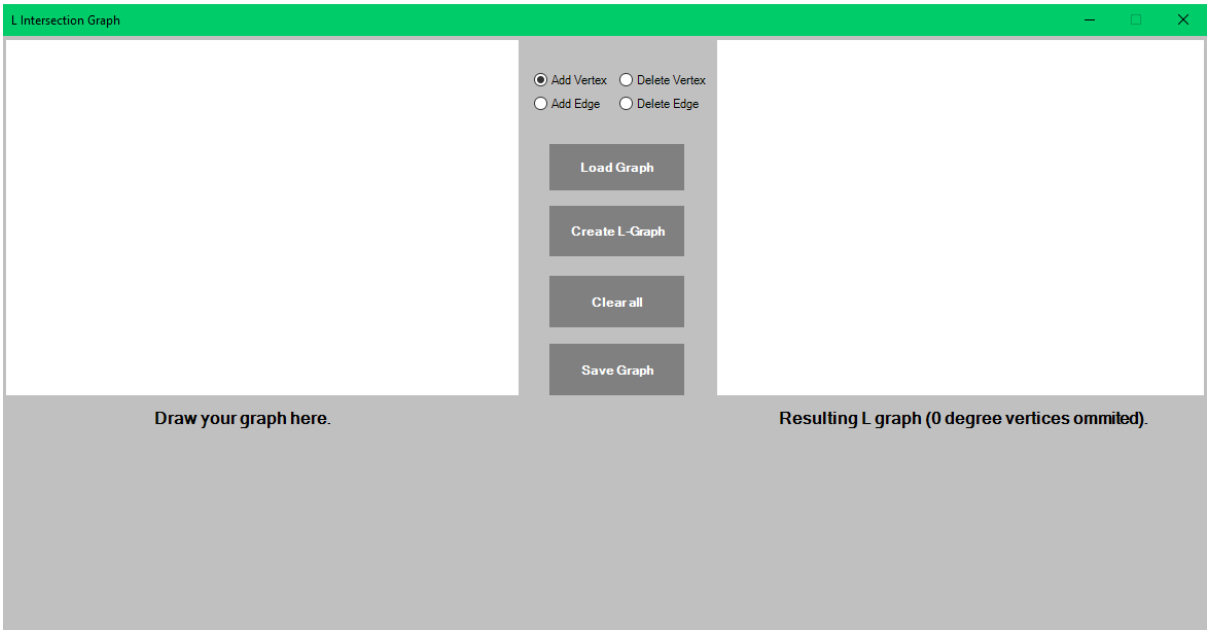
Příklad grafu nahrávaného ze souboru:

```
1 2
3 1
4 3
```

Druhou možností je graf naklikat do uživatelského rozhraní. Ke grafu nahranému ze souboru lze následně přidávat hrany i vrcholy. Vytvořené grafy je možné uložit tlačítkem `Save Graph`.

Pomocí tlačítka `Clear all` se smaže veškerý nahraný obsah.

L průniková reprezentace se vytvoří pomocí tlačítka `Create L-Graph`.



Obrázek 1: Uživatelské rozhraní

Programátorská část

Program je napsán v jazyce C++ a jeho funkcionalita je vytvořena v následujících souborech:

MyForm.{h,cpp,resx}

GraphLoader.{h,cpp}

VertexPermuter.{h,cpp}

Permutation.{h,cpp}

LIntersect.{h,cpp}

PartialOrder.{h,cpp}

LShape.h

MyForm

Definuje uživatelské rozhraní, jeho grafiku a funkcionalitu grafických prvků. Obsahuje instanci třídy GraphLoader.

GraphLoader

Zajišťuje načtení grafu do své proměnné `std::map<size_t, std::set<size_t>> neighbors`, v níž klíčem je ID vrcholu a hodnotou set jeho sousedů. Funkce

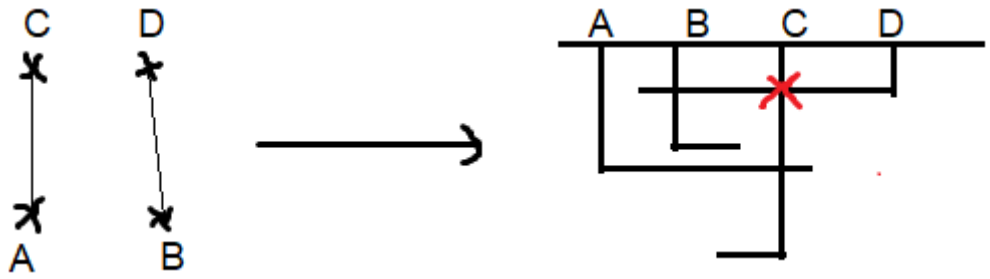
`removeZeroDegreeVertices` odstraní z `neighbors` vrcholy stupně nula. Funkce `stackByComponents` seřadí vrcholy tak, aby vrcholy každé souvislé komponenty byly seřazeny za sebou (tj. obě funkce mění vstupní ID vrcholů a seřazením je myšleno přejmenování tak, aby vrcholy v řadě 1..N splňovaly danou vlastnost). Proměnná `std::vector<size_t> vertexID` drží konverzi mezi původními a současnými ID. Na takto upravené mapě `neighbors` se volá funkce `permuteNeighbors`, která vrací `True` pokud byla nalezena L průniková reprezentace. Výsledná reprezentace je uložena v instanci třídy `LIntersectionGraph`, která je přístupná v `MyForm`, kde je kód zajišťující její vykreslení.

VertexPermuter

Pro každou souvislou komponentu vytvoří instanci třídy `Permutation`, které dodá vrcholy (jako `neighbors`) s ID nastavenými na 1..|komponenta| a jejich maticí sousednosti. Funkce této třídy, `permuteVertices`, je volána ve funkci `permuteNeighbors` třídy `GraphLoader`, kde vrací `True`, pokud byla nalezena L průniková reprezentace pro každou komponentu.

Permutation

Hledá vhodné permutace pořadí ukotvených vrcholů. Pro každou vhodnou permutaci volá funkci `createLGraph` třídy `LIntersectionGraph`, dokud tato funkce nevydá L průnikovou reprezentaci. Tato třída zmenšuje prostor prohledávaných permutací ukotvení vrcholů. Vhodné permutace se hledají rekurzivně od nejmenší permutace. Permutace se tvoří postupně od prvního ukotveného vrcholu a další vrchol se do permutace přidá, pouze pokud jeho přidáním nevznikne následující pro L průnikovou reprezentaci neřešitelná situace:



Obrázek 2: Nepřípustná situace pro L průnikovou reprezentaci

Tento konflikt, vznikající pokud se do permutace přidá vrchol C, se vyřeší zařazením vrcholu C za vrchol D, čímž vznikne nejmenší možná větší permutace a několik se jich přeskočí. Rekurze skončí pokud `createLGraph` vrátí řešení, nebo když se permutace projdou až k té největší.

LIntersect

Definuje třídu `LIntersectionGraph`, která pro vrcholy s ID 1..N hledá L průnikovou reprezentaci, přičemž ID vrcholů určují pořadí ukotvení vrcholů. L graf si lze představit na matici o rozměrech N x N, kde N je počet vrcholů. Tím, že je pořadí ukotvení vrcholů dané,

stačí každý L tvar popsat dvěma souřadnicemi: souřadnicí udávající, jak dlouhá je vertikální část L (tj., řádkovou souřadnicí bodu ohnutí L) a souřadnicí udávající, jak dlouhá je horizontální část L (tj., sloupcovou souřadnicí stranového vrcholu L). Protože pro protnutí dvou L tvarů je potřeba, aby byly v matici různě vysoko, stačí body ohnutí vybírat jako kombinaci 1..N. Hodnoty stranových vrcholů takové omezení nemají. V implementaci je navíc pro každý L tvar proměnná udávající, jestli L ukazuje vlevo, nebo vpravo.

Řešení se hledá rekurzivně ve dvou rovinách. První rovina je rovina směrů L tvarů. Nastavení tvaru vpravo, či vlevo může znamenat, že další tvary také musejí ukazovat tím, či oním směrem. Navíc pokud vrchol nemá sousedy s ID nižším, respektive vyšším než on sám, není třeba, aby ukazoval vlevo, respektive vpravo. Zbylé možnosti směrů se zkouší rekurzivně. Rekurze skončí, jakmile se najde průniková reprezentace. Druhá rovina je rovina souřadnic stranového vrcholu L. Podle nastavených směrů L tvarů se nastaví minimální hodnoty (minimální ve smyslu délky L, ne ve smyslu hodnoty na matici) těchto souřadnic. Dále se rekurzivně horizontální části L tvarů prodlužují, tak aby byly průnikem pokryty všechny hrany. Prodlužování probíhá tak, že se vyčerpají všechny možnosti prodloužení L tvarů, tj. prodlouží se nejdřív jeden a pak druhý tvar tvořící hranu, dokud není nalezeno řešení.

Tyto rekurze nezaručují, že bylo nalezeno řešení, protože sice pokrývají všechny průniky pro hrany, ale neřeší, že dva vrcholy bez hrany nesmějí mít průnik. Funkce `doPartialOrder` řeší tento problém tak, že se pokouší vytvořit částečné uspořádání vrcholů podle pozice bodu ohnutí L tvaru na matici. Rekurzemi získaný popis L tvaru spolu se znalostí sousedů příslušného vrcholu jasně určuje jeho pozici vůči ostatním v rámci částečného uspořádání. Pokud se při tvorbě částečného uspořádání nenajde spor, znamená to, že průniková reprezentace je validní a seřazení odpovídající částečnému uspořádání odpovídá sloupcovým souřadnicím bodů ohnutí L tvarů.

PartialOrder

Definuje částečné uspořádání využívané třídou `LIntersectionGraph`. Pomocí funkcí `add` a `transitivity` kontroluje validitu uspořádání uloženého v `std::vector<std::vector<int>> cum`. Funkce `createOrdering` vrací korektní uspoření (tj. permutaci).

LShape

Definuje L tvar pomocí tří souřadnic: horního bodu L, bodu ohnutí L a stranového bodu L.