###### Курсова робота

### З дисципліни” Основи дискретної математики”

На тему: **Алгоритм Дейкстра**

Зміст

|  |
| --- |
| 1.Вступ…………………………………………………………………………………..…………32.Елементи теорії графів:Основні визначення……………………………………………………………..…………..3Ізоморфізм, гомеоморфізм……………………………………………………….…………4 Шляхи і цикли…………………………………………………………………….…………5Дерева………………………………………………………………………………………..5Цикломатичне число і фундаментальні цикли……………….……….…………………..8Компланарні графи ………………………………………………………………..……….8Розфарбування графів………………………………………………………………….….10 Графи з атрибутами ……………………………………………………………….………12Незалежні безлічі і покриття ………………………………………………………..……12 3.Задача знаходження мінімального шляху в графах:Алгоритм Дейкстра…………………………………………………………….….………14Текст програми написаної на основі алгоритму Дейкстра………………………..…….15Результат виконання програми…………………………………………………………...17Графічне зображення початкового графа та дерево мінімальних шляхів після виконання рограми……………………………………………………………….……..…18**4.Висновок**………………………..……..……………………………………………………….18**5.** [**Література**](http://www.caravan.ru/~alexch/graphs/#Литература) ………………………………………………………………………..…….……..19 |

**1. Вступ**

 Останнім часом дослідження в областях, що традиційно відносяться до дискретної математики, займають усе більш помітне місце. Поряд з такими класичними розділами математики, як математичний аналіз, диференціальні рівняння, у навчальних планах спеціальності "Прикладна математика" і багатьох інших спеціальностей з'явилися розділи по математичній логіці, алгебрі, комбінаториці і теорії графів. Причини цього неважко зрозуміти, просто розглянувши задачу, розв'язувану пошуку найкоротшого шляху в графі .

2. Елементи теорії графів

Основні визначення

**Граф** (graph) - пари G=(V,E), де V - безліч об'єктів довільної природи, називаних **вершинами** (vertices, nodes), а E - сімейство пар ei=(vi1, vi2), vijV, називаних **ребрами** (edges). У загальному випадку безліч V і/чи сімейство E можуть містити нескінченне число елементів, але ми будемо розглядати тільки *кінцеві графи*, тобто графи, у яких як V, так і E кінцеві.

У приведеному визначенні графа E не випадково названо сімейством пар, а не безліччю. Справа в тім, що елементи E можуть бути не унікальні, тобто можливі **кратні ребра**. Існує інше, більш коректне визначення: граф визначається як трійка G=(V,E,), де V - безліч вершин, E - безліч ребер, а =(v,u,e) - тримісний предикат (булевська функція від трьох перемінних), що повертає True тоді і тільки тоді, коли ребро e інцидентне вершинам v і u. Однак такі "строгості" у нашому викладі є надмірними.

Якщо порядок елементів, що входять у ei, має значення, то граф називається **орієнтованим** (directed graph), скорочено - орграф (digraph), інакше - **неорієнтованим** (undirected graph). Ребра орграфа називаються **дугами** (arcs). Надалі будемо вважати, що термін "граф", застосовуваний без уточнень "орієнтований" чи "неорієнтований", позначає неорієнтований граф.

Приклад: G=(V,E); V={1,2,3,4}; E=<(1,1), (1,2), (1,3), (2,4), (2,4)>

  **G**
Якщо e=(v,u), те вершини v і u називаються **кінцями ребра**. При цьому говорять, що ребро e є **суміжним** (**інцидентним**) кожної з вершин v і u. Вершини v і u також називаються суміжними (інцидентними). У загальному випадку, допускаються ребра виду e=(v,v); такі ребра називаються **петлями**.

**Ступінь вершини** графа - це число ребер, інцидентних даній вершині, причому петлі враховуються двічі. Оскільки кожне ребро інцидентне двом вершинам, сума ступенів усіх вершин графа дорівнює подвоєній кількості ребер: Sum(deg(vi), i=1..|V|)=2|E|.

Граф, що не містить петель і кратних ребер, називається **звичайним**, чи **простим графом** (simple graph). У багатьох публікаціях використовується інша термінологія: під графом розуміється простий граф, граф із кратними ребрами називають **мультиграфом**, з петлями - **псевдографом**.

Деякі класи графів одержали особливі найменування. Граф з будь-якою кількістю вершин, не утримуючих ребер, називається **порожнім**. Звичайний граф з n вершинами, будь-яка пара вершин якого з'єднана ребром, називається **повним** і позначається Kn (очевидно, що в повному графі n(n-1)/2 ребер).

Граф, вершини якого можна розбити на непересічні підмножини V1 і V2 так, що ніякі дві вершини, що належать тому самому підмножині, не суміжні, називається **двочастковим** (чи біхроматичним, чи графом Кенига) і позначається Bmn (m=|V1|, n=|V2|, m+n=|V|). **Повний двочастковий** граф - такий двочастковий граф, що кожна вершина безлічі V1 зв'язана з усіма вершинами безлічі V2, і навпаки; позначення - Kmn. Зауваження: *повний двочастковий* граф Bmn не є *повним* (за винятком B11=K2).

 **B33**

**Підграфом**, чи **частиною** графа G=(V,E) називається такий граф G'=(V',E'), що V'V і дві несуміжні вершини в G не суміжні в G'. **Повним підграфом** називається підграф, будь-яка пара вершин якого суміжна.

**Основним підграфом (суграфом)** графа G називається будь-який його підграф, що містить ту ж безліч вершин, що і G.

Ізоморфізм, гомеоморфізм

Графи G1=(V1,E1) і G2=(V2,E2) називаються **ізоморфними** (позначення: G1~G2), якщо між графами існує взаємо-однозначне відображення : G1G2 (V1V2, E1E2), що зберігає відповідність між ребрами (дугами) графів, тобто для будь-якого ребра (дуги) e=(v,u) вірно: е'=(v,u)=((v),(u)) (eE1, е'E2). Відображення  називається **ізоморфним відображенням**.

Іншими словами, ізоморфні графи розрізняються тільки позначенням вершин.

Ізоморфні графи. Одне з ізоморфних відображень: (0,0), (1,3), (2,5), (3,6), (4,7), (5,2), (6,1), (7,4), (8,9), (9,8).

Характеристики графів, інваріантні відносно ізоморфизмов графів (тобто приймаючі однакові значення на ізоморфних графах), називаються інваріантами графів.

**Підрозділом ребра** (v1,v2) графи називається операція додавання в граф вершини v' і заміни цього ребра на два суміжних ребра (v1,v') і (v',v2): V'=V+{v'}, E'=E-{(v1,v2)}+{(v1,v')}+{(v',v2).

Граф G' називається **підрозділом графа** G, якщо він може бути отриманий з G шляхом кінцевого числа підрозділів ребер.

Дві графи називаються **гомеоморфними**, якщо для них існують ізоморфні підрозділи.

Шляхи і цикли

**Шляхом** у графі (чи **маршрутом** в орграфі) називається послідовність вершин, що чергується, і ребер (чи дуг - в орграфі) виду v0, (v0,v1), v1, ... , (vn-1,vn), vn. Число n називається **довжиною шляху**. Шлях без повторюваних ребер називається **ланцюгом**, без повторюваних вершин - **простим ланцюгом**. Шлях може бути замкнутим (v0=vn). Замкнутий шлях без повторюваних ребер називається **циклом** (чи **контуром** в орграфі); без повторюваних вершин (крім першої й останньої) - **простим циклом**.

**Твердження 1.** Якщо в графі існує шлях, що веде з вершини v0 у vn, то існує і простий ланцюг між цими вершинами.

Доказ: такий простий ланцюг можна побудувати, "викинувши" зі шляху всі цикли.

~

Граф називається **зв'язковим**, якщо існує шлях між будь-якими двома його вершинами, і **незв'язним** - у противному випадку. Незв'язний граф складається з декількох зв'язних **компонентів** (зв'язкових підграфов).

Для орграфів поняття св'язаність є більш складним: розрізняють сильну св'язаність, однобічну звязність і слабку зв’язність. Орграф називається **сильно зв'язковим**, якщо для будь-яких двох його вершин v і u існує як маршрут з v у u (v->u), так і з u у v (u->v). Орграф називається **односторонньо зв'язковим**, якщо для будь-яких двох його вершин u і v існує по крайньої один з маршрутів v->u чи u->v. Нарешті, орграф називається **слабко зв'язковим**, якщо зв'язний неорієнтований граф, одержуваний з цього орграфа шляхом зняття орієнтації з дуг. Очевидно, що будь-який сильно зв'язний граф є односторонньо зв'язковим, а односторонньо зв'язний - слабко зв'язковим, але не навпаки.

Дерева

**Деревом** називається довільний зв'язний граф без циклів.

**Лема 1.** Нехай G=(V,E) - зв'язний граф, вершини v1 і v2 якого не суміжні. Тоді в графі G'=(V,E+(v1,v2)) існує простий цикл, що проходить через ребро (v1,v2).

Доказ: тому що G - зв'язний, у ньому існує шлях з v2 і v1, а значить ([по утвержденю 1](http://www.caravan.ru/~alexch/graphs/#Утв1)),і простий ланцюг v2...v1. Отже, у графі G' існує шлях v2...v1(v1,v2)v2, що є простим циклом (по визначенню).

~

**Лема 2.** Нехай G=(V,E) - зв'язний граф, ребро e=(v1,v2) якого входить у деякий цикл. Тоді граф G'=(V,E-e) - також зв'язний, тобто при видаленні **кільцевого ребра** (ребра, що входить у деякий цикл) зі зв'язного графа цей граф залишається зв'язковим.

Доказ: тому що G - зв'язний, у ньому існує шлях S між будь-якими двома вершинами vi і vj. Якщо e не входить у шлях S=vi...vj, то цей шлях існує й у графі G', а виходить, G' залишається зв'язковим. Інакше (e входить у цей шлях): S=vi...v1(v1,v2)v2...vj. За умовою e - входить у деякий цикл, отже, існує замкнутий шлях C=v2(v2,v1)v1Tv2 (початком замкнутого шляху ми можемо вважати будь-яку його вершину), причому ребро e=(v1,v2) не входить у T (якщо існує шлях між вершинами, то існує і шлях, що є простим ланцюгом - див. [утвердження 1](http://www.caravan.ru/~alexch/graphs/#Утв1)). Але тоді існує шлях S'=vi...v1Tv2...vj, у котрій не входить ребро e=(v1,v2) і, отже, цей шлях існує в графі G'.

~

**Лема 3.** Нехай G=(V,E), p=|V|, q=|E|.
1) число зв'язних компонентів у G більше або дорівнює |V|-|E| (Nкомп.p-q);
2) якщо в G немає циклів, то число зв'язних компонентів у G дорівнює |V|-|E| (Nкомп.=p-q).

Доказ: побудуємо порожній граф з p вершинами (очевидно, у ньому рівно p зв'язкових компонент) і будемо додавати ребра по одному.

При додаванні ребра можливі дві ситуації: (а) нове ребро з'єднує вершини, що знаходилися до цього в різних компонентах (у цьому випадку кількість компонент зменшується на одиницю) і (б) нове ребро з'єднує вершини, що належать одному компоненту (число компонентів не змінюється). Отже, при додаванні q ребер число компонент зменшиться не більше ніж на q, і, отже, кількість компонентів у графі буде більше або дорівнює p-q. Це доводить твердження (1).

Відповідно до леми 1, при додаванні ребра в зв'язний граф у ньому з'являється цикл. Якщо в графі немає циклів, це означає, що при додаванні ребер завжди відбувався варіант (а) - інакше з'явилися б цикли. Отже, число компонентів при кожнім додаванні ребра зменшувалося на одиницю, і після додавання q ребер у графі буде рівно p-q компонент. Це доводить твердження (2).

~

**Наслідок 1 леми 3:** якщо |E||V|-2, те граф G=(V,E) незв'язний (випливає безпосередньо з [лемі 3](http://www.caravan.ru/~alexch/graphs/#Лемма3)).

**Теорема 1.** Любою зв'язний граф містить підграф, що є деревом.

Доказ: якщо в зв'язному графі немає циклів, то він уже є деревом по визначенню. Інакше знаходимо будь-як кільцеве ребро і видаляємо його; відповідно до [лемми 2](http://www.caravan.ru/~alexch/graphs/#Лемма2) граф залишається зв'язковим. Продовжуємо процес, поки в графі існують цикли. У силу кінцівки графа цей алгоритм побудує дерево за кінцеве число кроків.

Зауваження: фактично доведене більш сильне твердження - що будь-який зв'язний граф містить **основній підграф** (підграф з тією же кількістю вершин, що і сам граф), що є деревом.

~

**Теорема 2.** Для будь-якого дерева G=(V,E) вірно: |V|-|E|=1.

Доказ: по визначенню, у дереві немає циклів, отже, відповідно до леми 3 у ньому рівно |V|-|E| зв'язкових компонент. Але по визначенню дерево зв'язне, тобто складається з одного зв'язного компонента, тому |V|-|E|=1.

~

**Теорема 3.** Наступні властивості графів еквівалентні:

1. G=(V,E) - дерево;
2. будь-які дві вершини G з'єднані єдиним простим ланцюгом;
3. G - граф без циклів, у якого |E|=|V|-1;
4. G - зв'язний граф, у якого |E|=|V|-1;
5. G - зв'язний граф, але при видаленні будь-якого ребра він стає незв'язним;
6. G - граф без циклів, але при додаванні будь-якого ребра в ньому з'являється рівно один (з точністю до завдання початкової вершини і напрямку обходу) простий цикл.

Доказ: доведемо теорему в послідовності (1)<=>(2), (2)=>(3)=>(4)=>(5)=>(6)=>(1).

**(1)=>(2):** допустимо, що деякі дві вершини v1 і v2 графа G з'єднані, принаймні, двома різними простими ланцюгами L1=u1....uk, де u1=v1 і uk=v2, і L2=w1....wm, де w1=v1 і wm=v2. З того, що ланцюги є простими і різними, випливає, що існує число j, 1<j<min(k,m), таке, що uj-1wj-1, ujwj, ... , uj+a-1wj+b-1, uj+awj+b, де a1, b1. Отже, у G існує цикл ІЗ=uj-1(uj-1,uj)uj...uj+a(wj+b,wj+b-1)wj+b-1... wj(wj,wj-1)wj-1 (див. малюнок) - одержали протиріччя з (1).

**(2)=>(1):**
(а) граф G є зв'язковим по визначенню связаність (будь-які дві вершини графа з'єднані ланцюгом);
(б) допустимо, що в графі G існує цикл, що проходить через деяку вершину v: C=v(v,u1)u1....uk(uk,v)v. Але це означає, що між v і кожної з вершин ui існують, принаймні, два різних шляхи L1=v(v,u1)u1...ui-1(ui-1,ui)ui і L2=v(v,uk)uk...ui+1(ui+1,ui)ui (шляхи різні, тому що по визначенню в циклі відсутні повторювані ребра). У силу [утвердження 1](http://www.caravan.ru/~alexch/graphs/#Утв1) з цих шляхів можна "виділити" прості ланцюги, що також будуть різні (у L1і L2 немає співпадаючих ребер), - одержали протиріччя з (2).
З (а), (б) і визначення дерева випливає, що G є деревом. **(2)=>(3):** по [теорема 2](http://www.caravan.ru/~alexch/graphs/#Т2);
**(3)=>(4):** по [лемма 3](http://www.caravan.ru/~alexch/graphs/#Лемма3);
**(4)=>(5):** т.к. |E|=|V|-1, те після видалення ребра в новому графі буде |V|-2 ребер, і по [слідству 1 лемки 3](http://www.caravan.ru/~alexch/graphs/#Сл1лем3) цей граф буде незв'язним;
**(5)=>(6):**
(a) доведемо першу частину твердження (G - граф без циклів): допустимо, у G є цикли; але тоді при видаленні будь-якого кільцевого ребра він залишиться зв'язковим, що суперечить (4);
(б) доведемо другу частину твердження (при додаванні будь-якого ребра в G з'являється рівно один простий цикл): зі связаність графа і [лемма 1](http://www.caravan.ru/~alexch/graphs/#Лемма1) випливає, що при додаванні будь-якого ребра в G з'являється, як мінімум, один простий цикл; у силу (2) цей простий цикл єдиний (зворотне означало б, що в G існують вершини, з'єднані більш ніж одним простим ланцюгом);
**(6)=>(1):** допустимо, G - не дерево, тобто граф чи не зв'язний містить цикли. Циклів не може бути в силу (5а), тому залишається варіант: G - незв'язний і складається мінімум із двох компонентів. Але тоді при додаванні ребра між вершинами, що належать різним компонентам, цикли не утворяться, а це суперечить (5б).

~

**Основним деревом (кістяком)** зв'язного графа називається будь-який його основний підграф, що є деревом.

Існування основного підграфа для будь-якого зв'язного графа доводиться [теоремою 1](http://www.caravan.ru/~alexch/graphs/#Т1).

Загальне число основних дерев зв'язного графа може бути дуже велика. Так, для повного графа з n вершинами воно дорівнює nn-2 (без доказу).

Граф і два його основних дерева (вилучені ребра показані пунктиром).

Для довільного (можливо, незв'язного) графа G **основне дерево** визначається як будь-який його основний підграф, не утримуючих циклів і маючи стільки ж компонентів связаність, що і G.

Цикломатичне число і фундаментальні цикли

**Цикломатичрим числом** графа G=(V,E) називається з k зв'язковими компонентами називається число (G)=|E|-|V|+k.

**Фундаментальним циклом** графа G=(V,E) з основним деревом T=(V,E') називається простий цикл, одержуваний у результаті додавання в T одного з ребер G, не приналежного E'.

**Твердження 1.** Кількість фундаментальних циклів графа G=(V,E) при будь-якому фіксованому основним дереві T=(V,E') дорівнює цикломатичному числу G.

Доказ: відповідно до [лемма 3 п.2](http://www.caravan.ru/~alexch/graphs/#Лемма3), k=|V|-|E'|, отже, <кількість ребер G, не приналежних E'> = |E|-|E'| = |E|-(|V|-k) = (G). При додаванні кожного з цих ребер у T з'являється рівно один простий цикл у силу [теоремі 3 п.6](http://www.caravan.ru/~alexch/graphs/#Т3); всі одержувані при цьому прості цикли різні, тому що кожний з них містить принаймні одне унікальне ребро - те саме ребро G, не приналежне E', що було додано в дерево.

~

Компланарні графи

Зіставивши вершинам графа крапки на чи площині в просторі, а ребрам - лінії, що з'єднують крапки, що відповідають кінцям ребра, можна одержати діаграму - візуальне представлення даного графа.
Очевидно, що для будь-якого графа можна побудувати нескінченну кількість таких діаграм. Якщо на деякій діаграмі серед крапок, що відповідають вершинам графа, немає співпадаючих, а лінії, що відповідають ребрам графа, не мають загальних крапок (за винятком кінців), то ця діаграма називається **геометричною реалізацією** графа.

**Теорема 1.** Для будь-якого графа існує геометрична реалізація в тривимірному евклідовому просторі.

Доказ:

1. реалізуємо |V| крапок, що відповідають вершинам графа, на одній прямій;
2. проведемо через цю пряму |E| різних на півплощин;
3. реалізуємо кожне ребро у своїй на півплощині.

~

Виникає питання: чи будь-який граф можна реалізувати на площині? Відповідь - негативний. Геометричну реалізацію на площині допускають лише деякі графи; такі графи називаються **компланарними.**

Для наступного викладу нам знадобиться поняття **грані**. Неформально визначимо грань як частина площини, на які площина "розрізається" лініями геометричної реалізації графа. Завжди існує необмежена зовнішня грань.

 - 7 вершин, 8 ребер, 3 грані

**Формула Ейлера.** Для будь-якої геометричної реалізації графа G=(V,E) на площині вірно: p-q+r=2, де p=|V|, q=|E|, r - число граней (без доказу).

**Теорема 2.** Якщо в зв'язковому планарним графі немає циклів довжини менше k і k3, то qk(p-2)/(k-2), де p=|V|, q=|E|.

Доказ (не зовсім формальне): нехай граф реалізований на площині і при цьому вийшло r граней. Нехай qi - число сторін у i-й грані (див. малюнок). Кожне ребро є стороною двох граней, тому 2q=Sum(qi, i=1..r). По умови в графі немає циклів довжини менше k, але тоді qik (тому що сторони грані утворять цикл) і 2q=Sum(qi, i=1..r)kr. По формулі Эйлера r=2-p+q, отже, 2qk(2-p+q), з чого випливає доказувана нерівність.

- 8 ребер, 3 грані, 3+6+7=16 сторін
~

**Наслідок 1 теореми 2:** для будь-якого зв'язкового пленарного графа без петель і кратних ребер виконується нерівність: q3(p-2), де p=|V|, q=|E|.

Доказ: тому що за умовою в графі немає петель і кратних ребер, у ньому немає і циклів довжини менше 3, тому, підставляючи k=3 у нерівність [теоремі 2](http://www.caravan.ru/~alexch/graphs/#ПланарТ2), одержуємо: qk(p-2)/(k-2)=3(p-2).

~

**Теорема 3.** Граф K5 не компланарний.

Доказ: K5 зв'язний, у ньому немає петель і кратних ребер, але наслідок 1 теореми 2 не виконується - q=10>3(p-2)=9. Виходить, K5 не компланарний.

~

**Теорема 4.** Граф K33 не компланарний.

Зауваження: використання наслідку 1 теореми 2 тут не допоможе, тому що q=9<3(p-2)=12.

Доказ: у K33 немає циклів довжини менше 4, тому застосуємо нерівність [теоремі 2](http://www.caravan.ru/~alexch/graphs/#ПланарТ2) безпосередньо (при k=4): q=9>4(p-2)/2=8. Отже, K33 не компланарний.

~

**Теорема Понтрягіна-Куратовского (критерій компланарності графів).** Граф G планарин тоді і тільки тоді, коли він не містить підграфов, гомеоморфних K5 чи K33.

Доказ: необхідність випливає з тверджень 1-4 (див. нижче), а також з того факту, що графи K5 і K33 не компланарні (відповідно до теорем [3](http://www.caravan.ru/~alexch/graphs/#ПланарТ3) і [4](http://www.caravan.ru/~alexch/graphs/#ПланарТ4)).

**Твердження 1**: якщо графи U1 і U2 ізоморфні, то U1 компланарний тоді і тільки тоді, коли U2 компланарний.

Доказ: будь-яка геометрична реалізація U1 є геометричною реалізацією U2, і навпаки.

**Твердження 2**: будь-який підрозділ U' графа U компланарний тоді і тільки тоді, коли U компланарний.

Доказ:
(=>) граф U' компланарний, отже, існує його геометрична реалізація на площині R'. Побудуємо по R' плоску геометричну реалізацію R графа U. Для цього об'єднаємо всі лінії R', що відповідають ребрам U', отриманим у результаті виконання операцій підрозділу ребер. У силу існування R граф U є компланарним.
<=) граф U компланарний, отже, існує його геометрична реалізація на площині R. Побудуємо по R плоску геометричну реалізацію R' графа U'. Для цього повторимо в будь-якій послідовності операції підрозділу ребер, що привели до побудови U'. Після виконання кожної з цих операцій будемо розбивати лінію, що відповідає ребру, що підрозділяється, на двох ліній (розбивка можна робити в будь-якій крапці, що не збігається з кінцями лінії). У силу існування R' граф U' є компланарним.

**Твердження 3:** якщо графи U1 і U2 гомеоморфні, те U1 компланарний тоді і тільки тоді, коли U2 компланарний.

Доказ:
(=>) за умовою U1 і U2 гомеоморфні  {по визначенню}  існують їхні ізоморфні підрозділи U1' і U2'. За умовою граф U1 компланарний  {по утв.2}  граф U1' компланарний  {по утв.1}  ізоморфний йому граф U2' компланарний  {по утв.2}  граф U2 компланарний.
(<=) аналогічно.

**Твердження 4:** якщо підграф U' графа U не компланарний, те U також не компланарний.

Доказ: допустимо, що граф U компланарний. Отже, існує його плоска геометрична реалізація R. Але тоді можна побудувати плоску геометричну реалізацію R' графа U': для цього досить видалити з R крапки і лінії, що відповідають тим вершинам і ребрам U, яких немає в U'. З існування R' випливає, що U' компланарний - одержали протиріччя.

Достатність теореми доводиться набагато складніше (див., наприклад, [[3](http://www.caravan.ru/~alexch/graphs/#З87)]).

~

Існують і інші критерії компланарності графів [[3](http://www.caravan.ru/~alexch/graphs/#З87)].

Розфарбування графів

**Верховим розфарбуванням** (далі - просто розфарбуванням) графа називається відображення безлічі вершин графа на кінцеву безліч (безліч квітів); **n-розфарбування** графа - розфарбування з використанням n квітів. Розфарбування називається **правильної**, якщо ніякі дві вершини одного кольору не суміжні. Очевидно, що для графа без петель завжди існує правильне розфарбування в |V| квітів.

**Хроматичним числом** графа G називається мінімальне число n=(G), таке, що існує правильне n-розфарбування.

**Лема 1.** У будь-якому компланарному графі без петель і кратних ребер існує вершина ступеня не більш п'яти.

Доказ: допустимо, що ступеня усіх вершин перевершують 5. Тоді 2q=Sum(deg(vi), i=1..|V|)p і q3p. Але по [слідству 1 теореми 2](http://www.caravan.ru/~alexch/graphs/#1_2) повинне виконуватися нерівність q3(p-2)<3p - одержали протиріччя.

~

**Теорема про п'ять фарб.** Кожен компланарний граф без петель і кратних ребер є не більш ніж 5-хроматичним.

Доказ: (індукцією по числу вершин).

При p=1 твердження теореми вірно. Допустимо, що (\*) твердження вірне для всіх p<p0. Доведемо, що тоді воно вірно і для p0.

Розглянемо компланарний граф G без петель і кратних ребер з p0 вершинами; по лемі 1 у ньому є вершина v0 ступеня не більш 5. По припущенню індукції ([\*](http://www.caravan.ru/~alexch/graphs/#*1)) граф G', одержуваний видаленням з G вершини v0 (очевидно, також компланарний), може бути розфарбований не більш, ніж у 5 квітів. Нехай (\*\*) v1...vk, k=deg(v0) - усі вершини-сусіди вершини v0, розташовані по годинній стрілці відносно v0. Якщо в розфарбуванні вершин v1...vk використовується не більш 4-х квітів, то "пофарбуємо" вершину v0 у що залишився 5-й колір і одержимо правильне розфарбування.

Залишилося розглянути випадок, коли в розфарбуванні вершин v1...vk у графі G' використовується 5 квітів (k=5). Нехай ci - колір вершини vi (i=1..5). Розглянемо безліч A, що складається з вершини v1 і усіх вершин графа G, крім v0, у котрі можна дійти з v1 тільки по вершинах квітів c1 і c3. Можливі два випадки:

1. v3A;
2. v3A.

У першому випадку поміняємо кольору вершин безлічі A (c1c3); фарбування при цьому залишиться правильної. Дійсно, безліч A складається по визначенню з *усіх* вершин квітів c1 і c3, у котрі можна дійти з v1, тому серед вершин, суміжних вершинам, що належать A, немає вершин квітів c1 чи c3. Після заміни квітів вершин безлічі A вершина v1 одержить колір з3, отже, можна використовувати колір c1 для "фарбування" вершини v0 і одержати в такий спосіб правильне розфарбування графа G.

 Залишається другий випадок. З приналежності вершини v3 безлічі A випливає, що існує шлях з v1 у v3 (v1Sv3), що проходить тільки по вершинах квітів c1 і c3 (див. малюнок). Розглянемо цикл L=v1Sv3(v3,v0)v0(v0,v1)v1 і замкнуту криву, що відповідає цьому циклу в геометричній реалізації графа. Вершина v2 знаходиться усередині цієї замкнутої кривої, а v4 - зовні. Це випливає, по-перше, з того, що лінії, що відповідають ребрам графа в його геометричній реалізації, не можуть перетинатися (не вважаючи кінців), і, по-друге, з ([\*\*](http://www.caravan.ru/~alexch/graphs/#**1)).Визначимо безліч B, що складається з вершини v2 і усіх вершин графа G, крім v0, у котрі можна дійти з v2 тільки по вершинах квітів c2 і c4. Вершина v4 не належить B, оскільки будь-який шлях з v2 у v4 повинний проходити, принаймні, через одну вершину, що належить циклу L - тобто або через вершину v0, або через вершину, пофарбовану в кольори c1 чи c3. Отже, як і в першому випадку, можна поміняти кольору вершин безлічі B (c2c4) і "пофарбувати" v0 у c2.

~

**Теорема про чотири фарби.** Кожен компланарний граф без петель і кратних ребер є не більш ніж 4-хроматичним.

Проблема чотирьох фарб залишалася невирішеної протягом багатьох літ. Затверджується, що ця теорема була доведена за допомогою хитромудрих міркувань і комп'ютерної програми в 1976 році (Kenneth Appel and Wolfgang Haken. Every Planar Map is Four Colorable. Contemporary Mathematics 98, American Mathematical Society, 1980). Короткий виклад ідеї їхнього доказу мається в [[3](http://www.caravan.ru/~alexch/graphs/#З87)].

Графи з атрибутами

У багатьох випадках елементам (вершинам і ребрам) графа ставляться у відповідність різні дані - атрибути (мітки). Якщо як атрибути використовуються цілі чи речовинні числа, то такі графи називають **зваженими**. Фактично, зважений граф - це функція, визначена на графі. Як атрибути можуть виступати і нечислові дані, тому я буду називати графів з атрибутами **позначеними**, чи **атрибутованнями** (Графами-а-графами). Наприклад, структурні формули хімічних сполук представляються молекулярними графами - А-графами, вершини яких відповідають атомам хімічної структури, а ребра - валентним зв'язкам між атомами. Для вершин молекулярного графа визначений, принаймні, атрибут "номер атома в періодичній таблиці елементів", для ребер - "тип валентного зв'язку (одинарна, подвійна, потрійна й ін.)"; можуть використовуватися додаткові атрибути, наприклад, заряд атома.

Для графів з атрибутами можна ввести посилене визначення ізоморфізму: будемо вважати дві А-графи ізоморфними, якщо вони ізоморфні в звичайному змісті, і, крім того, ізоморфне відображення зберігає атрибути (тобто атрибути відповідних вершин і ребер в обох графах збігаються).

Незалежні безлічі і покриття

**Незалежна безліч вершин (НМВ)** - безліч вершин графа, ніякі дві вершини якого не інцидентні.

**Максимальна незалежна безліч вершин (МНМВ)** - НМВ, що не міститься ні в якому іншому НМВ.

Зауваження: у даному визначенні "максимальність" означає "нерозширюваність"; у загальному випадку граф може мати трохи МНМВ різної потужності.

**Найбільша незалежна безліч вершин** - НМВ максимальної потужності.

Потужність найбільшого НМВ (очевидно, це одне з МНМВ) називається **верховим числом незалежності** графа (а також **нещільністю**, **числом внутрішньої** чи стійкості **числом верхового упакування** графа); позначення - (G).

**Незалежна безліч ребер (НМР)**, чи **паросполучення** - безліч ребер графа, ніякі два ребра якого не інцидентні.

Потужність найбільшого паросполучення називається **числом паросполучення** графа; позначення - (G).

**Домінуюче безліч вершин (ДМВ)** - така безліч вершин графа, що кожна вершина графа або належить ДМВ, або інцидентна деякій вершині, що належить ДМВ.

**Верхове покриття (ВП)** - така безліч вершин графа, що кожне ребро графа інцидентне хоча б одній вершині, що належить ДМВ.

Потужність найменшого **верхового покриття** називається **числом верхового покриття** графа; позначення - (G).

**Домінуюче безліч ребер (ДМР)**, чи **реберне покриття** - така безліч ребер зв'язного графа, що кожна вершина графа інцидентна хоча б одному ребру, що входить у ДМР.

Потужність найменшого ДМР називається **числом реберного покриття** графа; позначення - (G).

На малюнку позначені реберне покритті графа (пунктиром), МНМВ (білі вершини) і верхове покриття (чорні вершини).

Величини (G), (G), (G) і (G) є інваріантами графа. Між цими інваріантами існує зв'язок, установлювана наступними лемами.

**Лема 1.** Безліч S є найменшим верховим покриттям графа G=(V,E) тоді і тільки тоді, коли T=V(G)\S є найбільшою незалежною безліччю вершин графа.

Доказ:
(=>)
1. доведемо, що ніякі дві вершини, що входять у T, не інцидентні (тобто T є НМВ). Допустимо зворотне: (vi,vj)E(G), viT, vjT. Але це означає, що ребро (vi,vj) не покривається безліччю S - протиріччя з визначенням ВП.
2. T є найбільшим НМВ у силу мінімальності |S| і тотожності |S| + |V(G)\S|  |V(G)|.
(<=)
1. доведемо, що кожне ребро G інцидентне хоча б одній вершині S (тобто S є ВП). Допустимо зворотне: (vi,vj)E(G), viS, vjS. Але це значить, що viT, vjT - протиріччя з визначенням НМВ.
2. аналогічно доказу (=>).

~

**Наслідок 1 леми 1.** Для будь-якого графа G=(V,E) сума числа верхового покриття і верхового числа незалежності постійна і дорівнює кількості вершин G: (G)+(G)=|V(G)|.

**Лема 2.** Якщо граф G=(V,E) не має ізольованих вершин, то сума його числа паросполучення і числа реберного покриття постійна і дорівнює кількості вершин G: (G)+(G)=|V(G)|.

Доказ:

1) Нехай C - найменше реберне покриття G, що містить (G) ребер. Розглянемо підграф GC графа G, утворений безліччю ребер C і інцидентними вершинами G; по визначенню покриття в нього входять усі вершини G. Оскільки C є найменшим, GC складається з однієї чи більшої кількості компонентів, кожна з який є деревом (дійсно, у противному випадку можна було б "викинути" з них кільцеві ребра й одержати покриття меншої потужності). По [теоремі 2](http://www.caravan.ru/~alexch/graphs/#Т2) кількість ребер у кожнім компоненті GSi графа GC на одиницю менше кількості вершин: |E(GCi)| = |V(GCi)| - 1. Просуммировав ці рівності для всіх i, одержимо: |E(GC)| = |V(GC)| - p, де p - кількість компонентів у GС, отже, p = |V(G)| - (G). З іншого боку, якщо взяти по одному ребру з кожного компонента GC, одержимо деяке паросполучення, отже, (G)  p = |V(G)| - (G) і (G) + (G)  |V(G)| (\*).

2) Нехай M - найбільше паросполучення G, що містить (G) ребер. Розглянемо безліч U вершин графа G, не покритих М. З визначення паросполучення випливає, що |U| = |V(G)| - 2|M| = |V(G)| - 2(G). Безліч U є незалежним (дійсно, якби дві довільні вершини U "зв'язувалися" ребром, то можна було б додати це ребро M і одержати паросполучення більшої потужності). Оскільки G за умовою не має ізольованих вершин, для кожної вершини, що входить у U, існує ребро, що покриває неї. Позначимо безліч таких ребер через S. Безлічі M і S не перетинаються, тому |M  S| = |M| + |S| = (G) + |U| = |V(G)| - (G). Об'єднання M і S є реберним покриттям графа по визначенню, отже, (G)  |M  S| = |V(G)| - (G) і (G) + (G)  |V(G)| (\*\*).

З нерівностей (\*) і (\*\*) випливає результат леми.

~

Подальші результати справедливі тільки для двочасткових графів.

**Теорема 1 (мінімаксная теорема Кеніга).** Якщо граф G є двочастковим, то (G) = (G).

(без доказу)

Визначення: **зроблене паросполучення (1-фактор)** - паросполучення, що покриває усі вершини графа.

Нехай X - довільна підмножина вершин графа G=(V,E). Позначимо через (X) безліч вершин G, інцидентних вершинам X.

**Теорема 2 (теорема про весілля).** Якщо G - двочастковий граф з частками P1 і P2, то G має зроблене паросполучення тоді і тільки тоді, коли |P1| = |P2| і, принаймні, одне з Pi (i=1..2) володіє тим властивістю, що для будь-якого X Pi виконується нерівність |X|  |(X)|.

(без доказу)

Назва теореми зв'язана з наступною "несерйозною" задачею: визначити, чи можливо "переженити" групу юнаків і дівчин так, щоб усі залишилися задоволені. Якщо допустити, що всі "симпатії" взаємні (припущення, прямо скажемо, нереалістичне), то задача зводиться до перебування зробленого паросполучення в двочастковому графі, вершини однієї з часток якого відповідають юнакам, іншої - дівчинам, і дві вершини зв'язані ребром тоді і тільки тоді, коли юнак і дівчина подобаються один одному.

**2.Задача знаходження мінмального шляху в графах:**

#####  Алгоритм Дейкстра

Розглянемо задачу про найкоротший шлях. Нехай G=(V,E) - зважений зв'язний граф з ненегативними вагами ребер (дуг). Вага f(e) ребра e інтерпретуємо як відстань між вершинами, суміжними даному ребру. Для заданої початкової вершини s і кінцевої вершини t шукається простий ланцюг, що з'єднує s і t мінімальної ваги. (s,t) - ланцюг мінімальної ваги називають найкоротшим (s,t) - шляхом. Очевидно, рішення задачі існує. Опишемо один з можливих алгоритмів рішення (Е. Дейкстра, 1959 р.).

Ініціалізація:

1. усім вершинам vi приписується мітка - речовинне число: d(s)=0, d(vi)=+ для всіх vis;
2. мітки усіх вершин, крім s, вважаються тимчасовими, мітка s - постійної;
3. вершина s з'являється поточної (c:=s);
4. усі ребра (дуги) вважаються непоміченими.

Основна частина:

1. для усіх вершин uj, інцидентних поточній вершині c, мітки яких є тимчасовими, перераховуємо ці мітки по формулі: d(uj):=min{d(uj), d(c)+Weight(c,uj)} (\*), де (c,uj) - ребро (дуга), що з'єднує вершини c і uj, а Weight(c,uj) - її вага; при наявності кратних ребер вибирається ребро з мінімальною вагою;
2. якщо мітки усіх вершин є постійними або рівні , те шлях не існує; ВИХІД("немає рішення");
3. інакше знаходимо серед вершин з тимчасовими мітками (серед усіх таких вершин, а не тільки тих, чиї мітки змінилися в результаті останнього виконання кроку (1)!) вершину x з мінімальною міткою, повідомляємо її мітку постійної, позначаємо ребро (дугу) (с',x), таке, що d(x) = d(с')+Weight(с',x), де с'=з або с' - вершина, що була поточної на одному з попередніх кроків (с'=з, якщо на кроці (1) при uj=x реалізувалася друга частина формули ([\*](http://www.caravan.ru/~alexch/graphs/#f1))), і робимо цю вершину поточної (c:=x);
4. якщо c=t, то знайдений шлях довжини d(t), якому можна відновити в такий спосіб: це той шлях між s і t, що складається тільки з позначених на кроці (3) ребер (дуг) (можна довести, що він існує й единий); ВИХІД("рішення знайдене");
5. інакше переходимо на крок (1).

Алгоритм можна модифікувати так, щоб він закінчував роботу після одержання усіма вершинами графа G постійних міток, тобто знаходяться найкоротші шляхи з s в усі вершини графа. Алгоритм визначить основне дерево D найкоротших шляхів з вершини s. Для будь-якої вершини v єдиний (s,v) - шлях у дереві D буде найкоротшим (s,v) - шляхом у графі G.

Алгоритм Дейкстра не завжди знаходить правильне рішення у випадку довільних ваг ребер (дуг) графа. Наприклад, для орграфа, зображеного на малюнку, алгоритм Дейкстра знайде маршрут s(s,t)t довжини 1 між вершинами s і t, а не мінімальний маршрут довжини 2+(-2)=0, що проходить через третю вершину графа.

 Приклад орграфа, для якого не будем застосовувати алгоритм Дейкста.


### Текст програми написаної на основі алгоритму Дейкстра

###

/\* Алгоритм пошуку дерева найкоротших шляхів у зваженому графі \*/

/\* Е.Дейкстра 1959 р. \*/

#include <stdio.h>

#include <stdlib.h>

#include <float.h>

int load\_matrix(int n, double\* weigh); /\* Уведення матриці ваг \*/

int deik(int n, int s, double\* weigh, int\* Q, double\* L); /\* Алгоритм пошуку \*/

void print(int n, int\* Q, double\* L); /\* Висновок результату \*/

void main(void){

 int n=0,s=0,ret=0;

 double \*weigh;

 int\* Q; /\* Масив міток покажчиків на попередню вершину \*/

 double\* L; /\* Масив найдених ваг шляху до вершини \*/

 printf("\nАлгоpитм пошуку дерева найкоротших шляхів у зваженому графі.\n");

 printf("Е.Дейкстpа, 1959 р.\n");

 printf("\nВведіть кількість вершин..");

 scanf("%d",&n);

 if (n <= 1){

 printf("\nКількість вершин повинне бути більше одиниці!\n");

 exit(1);

 }

 printf("\nВведіть початкову вершину..");

 scanf("%d",&s);

 s--;

 if ((s < 0)||(s > (n-1))){

 printf("\nПочаткова вершина зазначена неправильно! \n");

 exit(1);

 }

 Q=malloc(n\*sizeof(int));

 L=malloc(n\*sizeof(double));

 weigh=malloc(sizeof(double)\*n\*n);

 if ((weigh == NULL)||(Q == NULL)||(L == NULL)){

 printf("\nHедостатньо пам'яті для завантаження масиву! \n");

 exit(2);

 }

 ret=load\_matrix(n,weigh);

 if (ret != 0){

 printf("\nПомилка введення даних!\n");

 exit(5);

 }

 ret=deik(n,s,weigh,Q,L);

 if (ret != 0){

 switch (ret){

 case 1 :

 printf("\nГpаф не є зв'язаним!\n");

 exit(3);

 case 2 :

 printf("\nHедостаточно пам'яті для роботи!\n");

 exit(4);

 }

 }

 print(n,Q,L);

 free(weigh);

 free(Q);

 free(L);

}

int load\_matrix(int n, double\* weigh){

 int i,j,k;

 double tmp;

 for (i=0; i < n; i++){

 for (j=0; j < n; j++){

 weigh[n\*i+j]=(-1);

 }

 }

 printf("\nВведіть послідовно ваги ребер для зазначених чи вершин -1 для несуміжних.");

 for (i=0; i < n; i++){

 for (j=i+1; j < n; j++){

 printf("\nВеpшини %d і %d ",i+1,j+1);

 k=scanf("%lf",&tmp);

 if (k != 1){return(1);}

 weigh[i\*n+j]=tmp;

 }

 }

 return(0);

}

int deik(int n,int s, double\* weigh, int\* Q, double\* L){

 int i,j,k;

 int\* QI; /\* Масив індикаторів сталості міток покажчиків \*/

 double tmp;

 QI=calloc(n,sizeof(int));

 if (QI == NULL){return(2);}

 QI[s]=1;

 for (i=0; i < n; i++){

 Q[i]=(-1);

 L[i]=DBL\_MAX;

 }

 Q[s]=s;

 L[s]=0;

 weigh[n\*(n-1)+s]=0;

 for (k=0; k < n; k++){ /\* Основний цикл по вершинах \*/

 for (i=0; i < n; i++){ /\* Цикл по рядках матриці ваг \*/

 for (j=i+1; j < n; j++){ /\* Цикл по стовпцях матриці ваг \*/

 if ((QI[i]+QI[j] == 1)&&

 (QI[i]\*QI[j] == 0)&&

 (weigh[i\*n+j] != (-1.0))&&

 (((QI[i] == 1)&&((L[i]+weigh[i\*n+j]) < L[j]))||

 ((QI[j] == 1)&&((L[j]+weigh[i\*n+j]) < L[i])))){

 if (QI[i] == 1){

 L[j]=L[i]+weigh[i\*n+j];

 Q[j]=i;

 }

 else{

 L[i]=L[j]+weigh[i\*n+j];

 Q[i]=j;

 }

 }

 }

 }

 for (tmp=DBL\_MAX,i=0; i < n; i++){

 if ((tmp > L[i])&&(QI[i] == 0)){

 tmp=L[i];

 j=i;

 }

 }

 QI[j]=1;

 }

 free(QI);

 return(0);

}

void print(int n, int\* Q, double\* L){

 int i;

 printf("\n\nДеpево найкоротших шляхів:");

 for (i=0; i < n; i++){

 printf("\nВеpшина: %d Предок: %d Вага: %5.2lf",i+1,Q[i]+1,L[i]);

 }

}

**Результат виконання програми**

Алгоритм пошуку дерева найкоротших шляхів у зваженому графі.

Е.Дейкстра, 1959 р.

Уведіть кількість вершин.. 6

Уведіть початкову вершину.. 6

Уведіть послідовно ваги ребер для зазначених чи вершин -1 для несуміжних.

Вершини 1 і 2 2

Вершини 1 і 3 -1

Вершини 1 і 4 2

Вершини 1 і 5 -1

Вершини 1 і 6 5

Вершини 2 і 3 2

Вершини 2 і 4 -1

Вершини 2 і 5 2

Вершини 2 і 6 -1

Вершини 3 і 4 -1

Вершини 3 і 5 -1

Вершини 3 і 6 12

Вершини 4 і 5 1

Вершини 4 і 6 2

Вершини 5 і 6 5

Дерево найкоротших шляхів:

Вершина: 1 Предок: 4 Вага: 4.00

Вершина: 2 Предок: 5 Вага: 5.00

Вершина: 3 Предок: 2 Вага: 7.00

Вершина: 4 Предок: 6 Вага: 2.00

Вершина: 5 Предок: 4 Вага: 3.00

Вершина: 6 Предок: 6 Вага: 0.00

**Графічне зображення початкового графа та дерева мінімальних шляхів після виконання програми**

|  |  |
| --- | --- |
| Тестовий зв'язний зважений для алгоритму пошуку дерева шляхів з вершини 6 (Е. Дейкстра 1959р.)  | Рішення: дерево найкоротших шляхів з вершини 6 |



**4.Висновок**

Ця курсова робота показує що дискретна математика, поряд з такими класичними розділами математики, як математичний аналіз, диференціальні рівняння, у навчальних планах спеціальності "Прикладна математика" і різні розділи по математичній логіці, алгебрі, комбінаториці і теорії графів тісно пов’язані із сучасним програмуванням. Причини цього неважко зрозуміти, просто розглянувши задачу, у цій курсовій роботі, яка за допомогою алгоритму Е. Дейкстра має змогу пошуку найкоротшого шляху в графі .

5.Література

1. Зыков А.А. Теорія кінцевих графів. - Новосибірськ: Наука, 1969.
2. Харари Ф. Теорія графів. - М.: Світ, 1973.
3. Зыков А.А. Основи теорії графів. - М.: Наука, 1987.
4. Кристофидес Н. Теорія графів. Алгоритмічний підхід. - М.: Світ, 1978.
5. Майника Э. Алгоритми оптимізації на мережах і графах. - М.: Світ, 1981.
6. Ловас Л., Пламмер М. Прикладні задачі теорії графів. Теорія паросочетаний у математику, фізику, хімії. - М.: Світ, 1998.