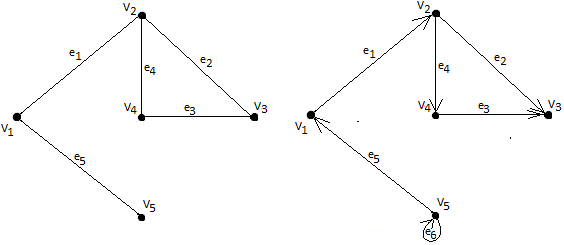
**Моделиране с графи. Примери за моделиране с графи. Представяне на графи.**

1. *Граф. Основни понятия*

Графът е съвкупност от две групи елементи – точки (върхове) и линии (ребра), съединяващи някои двойки точки. Най – често точките се изобразяват в равнина, а линиите са прави. Множеството на върховете се бележи с V, а множеството на ребрата с Е. Съвкупността *G(V,E)* ще наричаме граф. При това ако за всяко ребро има дефинирана посока (логическо „начало“ и „край“) се нарича ориентиран граф, в противен случай неориентиран. При ориентираните графи ребрата се наричат дъги.



*Фиг.1.1. Илюстрация на неориентиран и ориентиран граф*

На фиг.1.1 от ляво е изобразен неориентиран граф а от дясно ориентиран. Върховете *v1* и *v2* се наричат краища за реброто *e1* и съответно начален и краен връх за дъгата *e1*.

Някои основни понятия и определения при графите са:

* Съседни върхове – върхове свързани с общо ребро или дъга;
* Входящи и изходящи дъги – влизащи и излизащи дъги за връх *vi*;
* Инцидентни връх и дъга(ребро) – когато върха се явява начален или краен за дъгата (реброто).
* Съседни дъги (ребра) – две дъги (ребра) са съседни, ако са инцидентни с един и същи връх на графа
* Паралелни дъги (ребра) – когато два върха *vi ,vj* са съединени с повече от една дъга (ребро)
* Степен на връх *vi* – броят на инцидентните с върха *vi* ребра (дъги) и се бележи с *d(vi)*;
* Висящи върхове – върхове с *d(vi)=1;*
* Изолирани върхове – върхове с *d(vi)=0;*
* Примка – когато началният и крайният връх на една дъга съвпадат;
* Верига – Нека *v1,v2,…,vn,vn+1* е произволна последователност от върхове. Верига се нарича последователност от дъги *e1,e2,…,en*, за които *ei=( vi, v(i+1))* или *ei=(v(i+1), vi), i=1,2…n.* Върхът *v1* се нарича начален връх за веригата, а върхът *vn+1* – краен връх на веригата. Дължина на веригата се разбира броя участващи в нея дъги.
* Път – Верига за която *ei=( vi, v(i+1))* за всяко  *i=1,2,…,n,* се нарича път (т.е. последователност от ребра спазваща посоката на ориентирания граф, позволяващи свързване на несъседни върхове).Понятието дължина на път, начален и краен връх на пътя се определят както при веригата;
* Контур – верига при която началния и крайният връх съвпадат.
* Цикъл – път при която началния и крайния връх съвпадат;
* Прост граф – това е граф без паралелни ребра(дъги) и примки
* Пълен граф – при който за всяка двойка върхове съществува ребро инцидентно с тях.

1. *Представяне на графи в паметта на компютъра*

Представянето на графи може да се извърши по два основни начина чрез матрично представяне и чрез списък на съседство.

* 1. *Матрично представяне*
* Матрица на съседство

Нека *G(V,E)* е произволен ориентиран граф с n върха и без паралелни дъги. Матрицата на съседство *B =(bij)n x n* се нарича матрица чиито елементи се определят по следния начин.

* *bij = 1 ,* ако *(vi ,vj) Е;*
* *bij=0,* в противен случай.

В случай на неориентиран граф *bij = 1* тогава и само тогава когато съществува ребро между *vi* и *vj*. Създава се матрица по редовете и колоните на която се поставят номерата на върховете (фиг.2.1.).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| http://vista-2008.com/wp-content/uploads/custom/Grafi/graf2.jpg | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | **1** | **2** | **3** | **4** | **5** | | **1** | **0** | **1** | **1** | **0** | **0** | | **2** | **1** | **0** | **1** | **1** | **1** | | **3** | **1** | **1** | **0** | **1** | **1** | | **4** | **0** | **1** | **1** | **0** | **1** | | **5** | **0** | **1** | **1** | **1** | **0** | |

*Фиг.2.1.Представяне на граф чрез матрица на съседство.*

Ако в графа няма примки, по главния диагонал трябва да има само нули. Ако графът е неориентиран, матрицата е симетрична спрямо главния диагонал (както е показано на фиг.2.1.).

Предимството на представянето чрез матрица на съседство е, че се изисква постоянно време (само еднократен достъп до паметта) за определянето дали има или не дъга между два дадени върха. В случаите, когато в графа има паралелни дъги или ребра, представянето с матрица на съседство може да се модифицира подходящо, така че *bij* да съдържа броя на ребрата вместо единица. Недостатък на представянето с матрица на съседство е, че използваната памет е n² клетки, даже и в случаите, когато графът има твърде малко ребра. Затова това представяне е доста неикономично.

* Матрица на инцидентност

Нека *G(V,E)* е граф без примки с *n* върха и *m* дъги. Матрицата *A=(aij)n x m* се определя по следния начин:

* *aij=1*, ако *vi* е начален връх за дъгата *ej*;
* *aij=-1*, ако *vi* е краен връх за дъгата *ej*;
* *aij=0*, ако *vi* не е инцидентен с дъгата *ej*;

Редовете на матрицата на инцидентност се наричат вектори на инцидентност за графа *G*. Представянето чрез матрица на инцидентност е показано на фиг.2.2.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | ***e1*** | ***e2*** | ***e3*** | ***e4*** | ***e5*** | | ***v1*** | **1** | **0** | **1** | **0** | **1** | | ***v2*** | **-1** | **1** | **0** | **0** | **0** | | ***v3*** | **0** | **-1** | **0** | **-1** | **-1** | | ***v4*** | **0** | **0** | **-1** | **1** | **0** | |

*Фиг.2.2. Представяне на граф чрез матрица на инцидентност*

Ако графа е неориентиран матрицата на инцидентност се дефинира аналогично както за ориентирани, с тази разлика, че всички елементи (-1) се заменят с (+1).

* Матрица на достижимост и контра-достижимост

Матрицата на достижимост *R=(rij)nxm* се определя по следния начин:

* *rij=1*, ако върха *vj* е достижим от *vi*;
* *rij=0*, в противен случай.

*R(vi)* е множеството на върховете в графа *G* достижими от дадения връх *vi*. Ако намерим за всеки връх *vi* съответното множество *R(vi)*, матрицата *R* лесно ще се построи *rij = 1*, когато *vj  R(vi)* и 0 в противен случай.

Може да се съобрази че : , където *Г(vi)* е множеството от върховете *vj*, за които в графа съществува дъга *(vi,vj),* т.е. върховете, достижими от *vi* чрез път с дължина *1*.

*Г(Г(vi))=Г2(vi)* ще бъде множеството от върхове, достижими от *vi* чрез път с дължина 2 и т.н., *Гs(vi)* ще бъде множеството върхове, достижими от *vi* с използването на път, чиято дължина е *s*.

Следва че по формулата по горе, изпълнявайки от ляво на дясно операцията обединение, ще намерим всички достижими от *vi* върхове, като ще извършване тази операция докато намереното множество не престане да нараства. На фиг.2.3 е показано представянето чрез матрица на достижимост.

|  |  |
| --- | --- |
| фиг 12 |  |

*Фиг.2.3. Представяне на граф чрез матрица н достижимост.*

Матрицата на контра-достижимост *Q* е транспонирана на матрицата за достижимост. *Q = (qij)n x n*, се определя по следния начин:

* *qij=1*, ако от върха *vj* е достижим върха *vi*
* *qij=0*, в противен случай.

Аналогично се определя множеството *Q(vi)* като множество върхове на графа *G*, такива че от всеки връх на това множество може да се достигне *vi*.

И в този случай операцията обединение се изпълнява от ляво на дясно докато не спре да се актуализира текущото множество *Q(vi)*.

Матрицата на контра-достижимост за графа на фиг2.3. представлява транспонираната матрица на достижимост т.е. *Q=RT*.

* Матрица (вектор) на степените.

Нека *G(V,E)* е прост граф. Негота матрица на степените D представляваща още вектор се определя по следния начин:

*D=(d(vi))1 x n=( d(v1), d(v2)*,…, *d(vn)).*

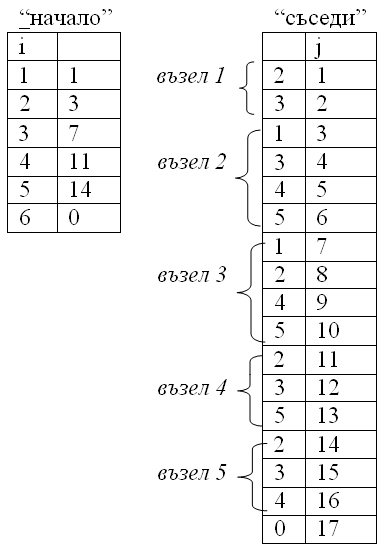
Елементите на вектора са неотрицателни цели числа. За пример на вектора на степените е използван графа от фиг.2. Неговия вектор е следния D=(2,4,4,3,3).

* 1. *Списък на съседите.*

Алтернативно представяне на граф *G(V,E)* е посредством списъци на съседство. Казваме, че връх *j* е съседен на връх *i*, когато има дъга от *i* до *j*. За представянето на графа използваме масив от *n* елемента, в който *i*-тия елемент указва списъка от съседи (в произволен ред) на върха *i*. Следователно необходимата памет за представяне на граф с *n* върха и *m* ребра е  от порядъка на *n+m*. Така се избягва недостатъка от използването на повече памет, отколкото е необходима. Недостатък на представянето чрез списъци на съседство е, че определянето дали има ребро от *i*до *j* може да изисква до *n* стъпки, защото е свързано с последователно обхождане на списъка от съседи на връх *i*. При списъците на съседство са възможни 3 подхода: статично(два масива), полудинамично (масив + списъци) и динамично (списъчни структури).

* Статично представяне

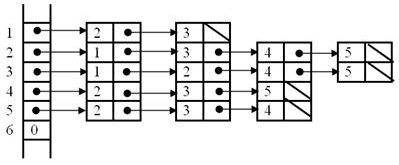
Използват се два масива: масив “съседи”, където последователно се изреждат номерата на съседите на всеки един възел в графа. Масив ”начало” - във всеки елемент се записва индекса, от който в масив “съседи” започва изреждането на съседите на възела, който има номер, равен на индекса на елемента в масив ”начало”. За примера на фиг.2.4 е използван графа от фиг.2.1.

****

*Фиг.2.4.Статично представяне на съседите от графа на фиг.2.1.*

* Полудинамично представяне

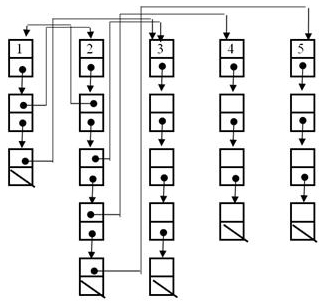
В масива “начало” се записва адресът на записа на първия съсед. Всеки запис на съсед притежава по две полета: номер на възела и указател към следващия съсед. Указателят за следващ съсед на последния от списъка е нула. За пример на фиг.2.5 отново е използван графа на фиг.2.1.



*Фиг.2.5. Полудинамично представяне на съседите от графа на фиг.2.1.*

* Динамично представяне.

За всеки възел се създава запис с две полета:  номер на възела и указател към двойка указатели. Първият указател от двойката съдържа ‘ а втория- адреса на следващата двойка указатели. На фиг.2.6. е даден пример за динамично представяне, като отново е използван графа на фиг.2.1.



*Фиг.2.6. Динамично представяне на съседите от графа на фиг.2.1.*

1. *Моделиране с графи. Примери за моделиране с графи.*

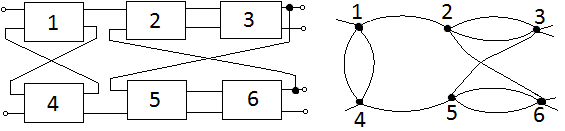
Графите намират широко приложение в решаването на най – различни задачи. Чрез тях се представя необходимата информация за решаването на тези задачи. В процеса на проектиране схемите и системите се представят чрез графи. Това може да стане с някои от следните начини:

* 1. *Представяне да схема (система) чрез граф.*

Ако множеството на върховете съответства множеството на елементите, а множеството на ребрата – връзките между елементите, то ел. схема се представя чрез граф.

*G(V,E),* като *V={v1,v2,…,vn} –* съответства множеството на елементите;

*E={e1,e2,…,en} –* съответства множеството на връзките м/у елементите*.*



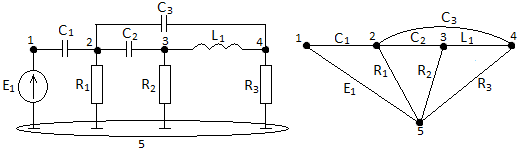
*Фиг.3.1.Представяне на схема чрез граф*

* 1. *Представяне на ел. схема чрез граф*

Ако на множеството на възлите на схемата се съпостави множеството на върховете *V* на графа *G(V,E),* а на множеството на елементите на схемата - множеството на ребрата *E,* то ел. схема се представя чрез граф.

*G(V,E),* като *V={v1,v2,…,vn} –* съответства множеството на възлите на сх.;

*E={e1,e2,…,en} –* съответства множеството на елементите*.*

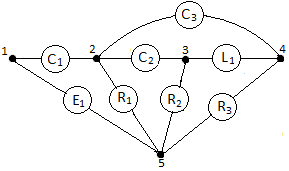


*Фиг.3.2. Представяне на ел. схема чрез граф.*

Това представяне може да стане и с друг вид граф в който върховете му се състоят от две множества:

* Върхове от първи вид – съответстващи на елементите на схемата
* Върхове от втори вид – съответстващи на възлите на схемата

Ребрата на този граф съединяват само върхове от различен вид. Представянето на схемата от фиг.3.2. по този начин ще има вида показан на фиг.3.3.



*Фиг.3.3. Представяне на ел. схема от фиг.3.2. чрез граф с два вида върхове.*

* 1. *Представяне на цифрова схема чрез граф.*

Нека е дадена система логически уравнения *xi=fi(x1,…,xn,xn+1,…,xn+q)*, където *xi(i=1,…,n+q)* са логически променливи, от които входни са тези с индекси *i=(n+1,…,n+q)*. Описание на една цифрова схема чрез граф може да се извърши, като всеки връх на графа се асоциира с променлива *xi*. Ориентирано ребро между две променливи с посока от *xi* към *xj* ще съществува, ако променливата *xi* е аргумент на *fi*, реализираща променливата *xj*.



*Фиг.3.4. Представяне на цифрова схема чрез граф.*