Моделирование поиска связных компонент графа средствами систем компьютерной алгебры

Васильев В.В., Тихомиров В.Г.

Тамбовский государственный технический университет

Элементы теории графов в литературе часто рассматриваются абстрактно без приведения алгоритмов решения задач. При этом обходятся стороной прикладные аспекты, в то время как, теория графов находит применение, например, при проектировании домов и сооружений, инженерных сетей, линий электропередач и т.п.

В связи с этим нами разработан комплекс учебных заданий, на основе которых составлены компьютерные программы, генерирующие различные варианты заданий: нахождения прямого и обратного транзитивных замыканий, компонент связности; топологической сортировки графа; поиск кратчайшего пути от вершины нижнего уровня до вершины верхнего уровня и его длины; нахождение центра графа и отклонения вершин графа от его центра, нахождения количества путей с заданным количеством дуг и самих путей; нахождения дополнительного графа, двойственного графа, реберного графа, нахождения остовов и базисных циклов.

С учетом изложенного ранее, нами выделено оптимальное количество учебных заданий, вокруг которых формируются дидактические единицы, приводящие к наибольшей эффективности усвоения содержания рассматриваемого курса. В результате формируется та часть учебной программы, где фиксируются: роль и место раздела «Теория графов» в курсе математики, основные цели его изучения, содержание учебного материала, технологии организации обучения.

В качестве примера приведем задание, состоящее в следующем: даны дуги орграфа, вершины которого пронумерованы от 1 до 16, причем первое число указывает начало, второе – конец дуги. На основе аналитических выражений для прямого и обратного транзитивных замыканий найти все компоненты связности для графа вашего варианта. Результаты вычислений проверить путем непосредственных преобразований матриц смежности.

Приведем основной фрагмент программы на языке «Maple». Существенно, что программы позволяют не только сгенерировать задание, но проверить ответ и ход решения, поскольку выдают не только ответ, но и результаты промежуточных вычислений.

```
 A:=RandomMatrix(n, density=0.5, generator=rand(0..1)):   G:=Graph(A):
TexMatrix(A, task):
СтроитьГраф(G, `1.bmp`);
 V:={$ 1..n}: vp:=[[0, 0] $ n ]:   l:=0:
 while V<>{}
 do
 B:={V[1]};
   writeline(sols, cat(` Находим компоненту связности для вершины `, tex(op(B)), `:`)):
   writeline(sols, cat(` Находим степени оператора прямого транзитивного замыкания для вершины `, tex(op(B)), `:`)):
   C1:=Замыкание(A, B, n, 1, sols):
   writeline(sols, cat(` Находим степени оператора обратного транзитивного замыкания для вершины `, tex(op(B)), `:`)):
   C2:=Замыкание(A, B, n, -1, sols):
    l:=l+1:
    C3:=C1 intersect C2:
    writeline(sols, cat(`Таким образом, компонента связности для вершины `, tex(op(B)), ` будет $ C_`, tex(op(B)), ` = `,
tex(C3), `$.` )):
     C[l]:=C3 union B:
    V:=V minus C[l]:
   od:
 for i from 1 to l
 do    placevertex(C[i], i, nops(C[i]))     od:
 SetVertexPositions(G, vp):
 СтроитьГраф(G, `2.bmp`);  # записать изображение графа G в файл
 print("vp=",vp):
```

Вначале с помощью генератора случайных чисел, распределенных равномерно, формируется матрица смежности графа. Затем с помощью процедуры «Замыкание» находятся прямое и обратное транзитивное замыкания выбранной вершины. После чего по найденным замыканиям вычисляется компонента связности текущей вершины. В соответствии с поставленным заданием процедура placevertex группирует вершины по компонентам.

Отметим, что процедура «Замыкание» не является стандартной и осуществляет поиск прямого и обратного транзитивных замыканий вершины.