УДК 004.4

ИСПОЛЬЗОВАНИЕ ПАКЕТОВ ПРИКЛАДНЫХ ПРОГРАММ ДЛЯ РЕАЛИЗАЦИИ СЕТЕВЫХ МЕТОДОВ ПЛАНИРОВАНИЯ

канд. техн. наук, доц. В.Л. ШАРСТНЕВ, Е.Ю. ВАРДОМАЦКАЯ
(Витебский государственный технологический университет)

Описан новый алгоритм решения задачи сетевого планирования средствами современных компьютерных технологий. В качестве инструмента решения использован пакет символьной математики MuPAD.

Сетевое планирование – это комплекс графических и расчетных методов организационных мероприятий, обеспечивающих моделирование, анализ и динамическую перестройку плана выполнения сложных проектов и разработок, например, таких как [1]:

- строительство и реконструкция каких-либо объектов;
- выполнение научно-исследовательских и конструкторских работ;
- подготовка производства к выпуску продукции;
- перевооружение армии.

Характерной особенностью таких проектов является то, что они состоят из ряда отдельных элементарных работ. Они обусловливают друг друга так, что выполнение некоторых работ не может быть начато раньше, чем завершены некоторые другие.

Классическое представление задачи сетевого планирования представлено на рисунке 1.

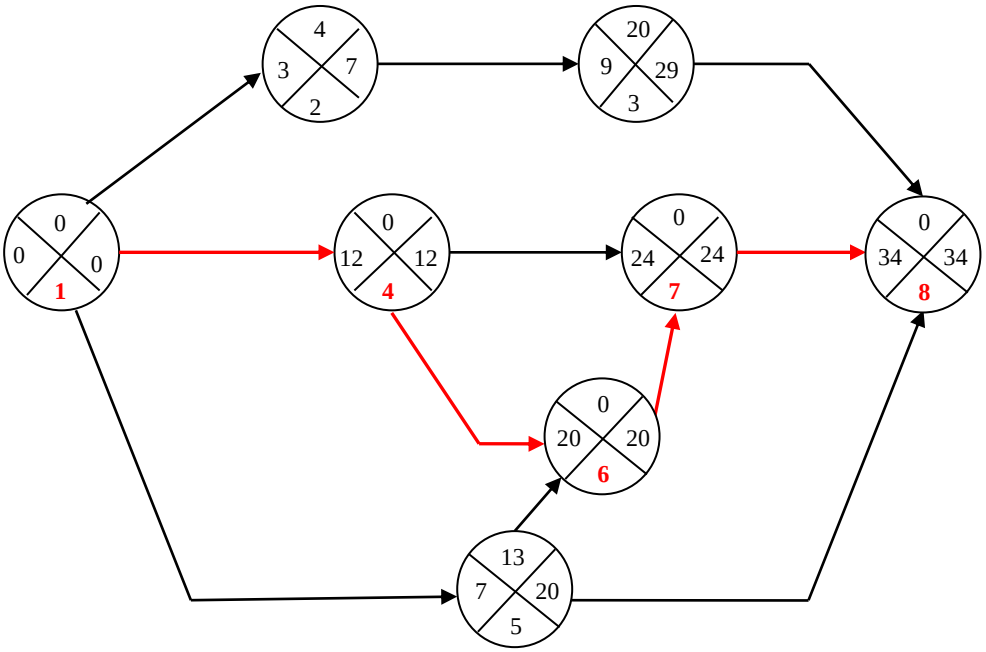

Рис. 1. Задача сетевого планирования

Основными понятиями сетевых моделей являются понятия работы, события и пути [2].

Работа – это некоторый процесс, приводящий к достижению определенного результата, требующий затрат каких-либо ресурсов и имеющий протяженность во времени. Работы на сетевом графике обозначаются стрелками, около которых ставится среднее время выполнения соответствующей работы.

Событие – это момент времени, когда завершаются одни работы и начинаются другие. При построении сетевого графика имеют место следующие события:

- исходное событие – это событие, в отношении которого предполагается, что оно не имеет предшествующей работы;
- завершающее событие – это событие, в отношении которого предполагается, что оно не имеет последующих работ;
- промежуточное или простое событие – это событие, характеризующее собой факт окончания всех предшествующих работ и начало всех последующих работ.

Событие обозначается кружком, который содержит следующую информацию (рис. 2).

Путь – это любая последовательность работ в сетевом графике, в которой конечное событие одной работы совпадает с начальным событием следующей за ней работы.

Различают:

- полный путь – путь от исходного до завершающего события;
- критический путь – максимальный по продолжительности полный путь;
- подкритический путь – полный путь, ближайший по длительности к критическому пути.

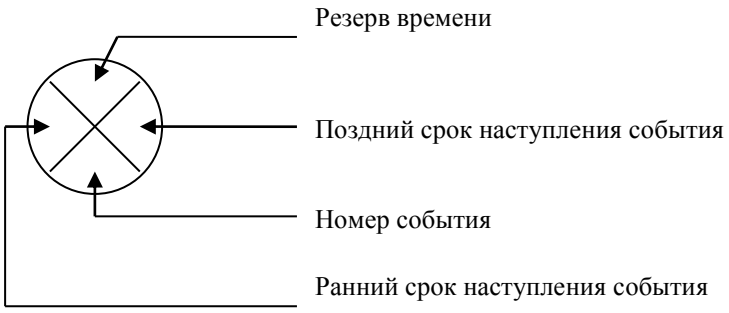

Рис. 2. Информация события

Расчет сетевого графика включает в себя расчет ранних и поздних сроков наступления событий, резервов времени каждого события и затем определение критического пути [3].

Путем последовательного перехода от исходного события, ранний срок свершения которого равен нулю, к завершающему событию рассчитываются ранние сроки его свершения.

Ранний срок наступления события представляет собой минимальный из возможных моментов наступления должного события при заданной продолжительности работ и начальном моменте. Ранний срок наступления j-го события T^p_j вычисляется по формуле:

$$T_j^p = max\left\{T_i^p + \overline{t}_{ij}\right\}, \qquad i = 1, \ldots k,$$

где $T_i^p (i = 1, \ldots k)$ – ранний срок наступления i-го события; $\overline{t}_{ij} (i = 1, \ldots k)$ – средняя продолжительность работы ij; k – число работ, непосредственно предшествующих j-му событию (все эти работы на сетевом графике обозначаются стрелками, входящими в кружок, обозначающий j-е событие).

Путем последовательного перехода от завершающего события, поздний срок которого равен величине критического пути, рассчитывают поздний срок его свершения. Этот срок T_j^n определяется разностью продолжительности критического пути и максимальным из путей, следующим за этим событием:

$$T_i^n = min\left\{T_i^n - \overline{t}_{ij}\right\}, \qquad j = 1, \ldots e,$$

где $T_i^n (j = 1, \ldots e)$ – поздний срок поступления j-го события; l – число работ, непосредственно следующих за i-м событием (все эти работы на сетевом графике обозначаются стрелками, выходящими из кружка, обозначающего i-е событие).

Разность между поздним и ранним сроком свершения событий – есть резерв времени этого события. Резерв времени i-го события R_i вычисляется по формуле:

$$R_i = T_i^n - T_j^p .$$

После вычисления резервов времени определяется критический путь $\alpha_{кр}$, т.е. полный путь, имеющий небольшую продолжительность. Для него является характерным, что все события, принадлежащие ему, не имеют резервов времени.

Для определения критического пути берутся все полные пути, проходящие через события с нулевым резервом времени. Затем подсчитывается их длительность и выбирается среди них путь, имеющий наибольшую продолжительность. Он и будет критическим.

В рассматриваемом примере критический путь проходит через события 1 – 4 – 6 – 7 – 8 и его длительность равна 34 дням.

Современные информационные технологии позволяют использовать некоторые другие подходы в решении приведенного выше типа задач. Можно достаточно легко и просто, опираясь на пакеты символьной математики, получить наглядное и точное решение. При этом совершенно не обязательно владеть алгоритмом решения. Достаточно четко и грамотно провести постановку задачи, подготовить данные и обработать их с использованием стандартных процедур [4].

Анализ пакетов символьной математики с целью использования для решения задач сетевого планирования показал, что самым оптимальным является наличие модулей по теории графов.

Принципиально любую задачу сетевого планирования можно свести к решению задачи по теории графов.

Покажем, как стандартную задачу сетевого планирования решить с использованием системы символьной математики MuPAD Pro 3.1 [5].

1. Зададим вершины (события) Vert исходного графа:

```
Vert:=[c1, c2, c3, c4, c5, c6, c7, c8]
```

```
[c1, c2, c3, c4, c5, c6, c7, c8]
```

2. Зададим ребра (работы) Ed исходного графа:

```
Ed:=[[c1, c2], [c1, c4], [c1, c5], [c2, c4], [c4, c6], [c5, c6], [c4, c7],
[c6, c7],[c2, c3], [c3, c8], [c7, c8], [c5, c8]]
```

```
[[c1, c2], [c1, c4], [c1, c5], [c2, c4], [c4, c6], [c5, c6], [c4, c7], [c6, c7], [c2, c3], [c3, c8], [c7, c8], [c5, c8]]
```

3. Зададим размер ребер (продолжительность работ) EdW исходного графа:

```
EdW:=[3,12,7,5,8,0,10,4,6,5,10,10]
```

```
[3, 12, 7, 5, 8, 0, 10, 4, 6, 5, 10, 10]
```

4. Зададим необязательное описание ребер (работ):

```
EdD:=["Работа 1-2", "Работа 1-4",
 "Работа 1-5", "Работа 2-4", "Работа 4-6", "Работа 5-6", "Работа 4-7",
"Работа 6-7", "Работа 2-3", "Работа 3-8", "Работа 7-8", "Работа 5-8"]
```

5. Сформируем исходный направленный граф (ключевое слово – Directed):

```
G := Graph(Vert, Ed, EdgeWeights = EdW, EdgeDescriptions = EdD, Directed):
```

6. Прорисуем исходный граф (рис. 3):

```
plot(Graph::plotCircleGraph(G,EdgeColor = RGB::Green), Footer="Исходный граф")
```

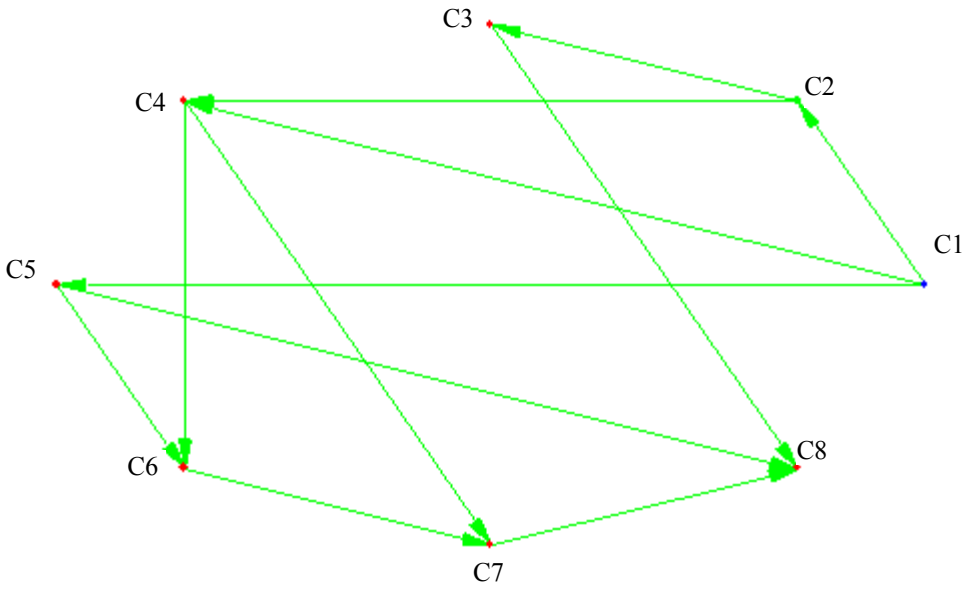

Рис. 3. Исходный граф

7. При необходимости вывод информации об исходном графе:

```
• Graph::printGraphInformation(G):
Vertices: [c1, c2, c3, c4, c5, c6, c7, c8]
Edges: [[c1, c2], [c1, c4], [c1, c5], [c2, c3], [c2, c4], [c3, c8],
Vertex weights: no vertex weights.
Edge descriptions: [c1, c2] = "Работа 1-2", [c1, c4] = "Работа 1-4"
 4-6", [c5, c6] = "Работа 5-6", [c4, c7] = "Работа 4-7", [c6, c7] =
c8] = "Работа 7-8", [c5, c8] = "Работа 5-8"
Edge weights: [c1, c2] = 3, [c1, c4] = 12, [c1, c5] = 7, [c2, c4] =
= 6, [c3, c8] = 5, [c7, c8] = 10, [c5, c8] = 10 (other existing edg
Edge costs: no edge costs.
Adjacency list (out): c1 = [c2, c4, c5], c2 = [c3, c4], c3 = [c8],
Adjacency list (in): c1 = [], c2 = [c1], c3 = [c2], c4 = [c1, c2],
Graph is directed.
```

8. Определим все возможные критические пути относительно исходного события c1:

```
• Temp:= Graph::longestPath(G, c1, Length, Path)
```

$$\begin{bmatrix} c1 = 0 \\ c2 = 3 \\ c3 = 9 \\ c4 = 12 \\ c5 = 7 \\ c6 = 20 \\ c7 = 24 \\ c8 = 34 \end{bmatrix}, \begin{bmatrix} c2 = [c2, c1] \\ c3 = [c3, c2, c1] \\ c4 = [c4, c1] \\ c5 = [c5, c1] \\ c6 = [c6, c4, c1] \\ c7 = [c7, c6, c4, c1] \\ c8 = [c8, c7, c6, c4, c1] \end{bmatrix}$$

Информативный вывод данных позволяет сделать заключение о том, что длительность критического пути «c1 – c8» составляет 34 дня. События, относящиеся к критическому пути, представляют собой последовательность «c1 – c4 – c6 – c7 – c8».

Выделим значение длительности критического пути:

```
• CPV:=CriticalPathValue:=op(Temp[1],nops(Temp[1]))
```

$$c8 = 34$$

и события, относящиеся к критическому пути:

```
• CP:=CriticalPath:=op(Temp[2],nops(Temp[2]))
```

$$c8 = [c8, c7, c6, c4, c1]$$

Принципиально задача об отыскании критического пути решена. Однако с целью наглядности покажем, как график критического пути отображается на исходном графе.

9. Создадим матрицу событий, относящихся к критическому пути:

```
• x:=matrix(nops(CP[2])-1,2):
• for i from nops(CP[2])-1 downto 1 do
    for j from 1 to 2 do
      if j=1 then x[i,j]:=op(CP[2],i+1)
             else x[i,j]:=op(CP[2],i)
      end_if:
    end_for:
  end_for:
• x
```

$$\begin{pmatrix} c7 & c8 \\ c6 & c7 \\ c4 & c6 \\ c1 & c4 \end{pmatrix}$$

10. Преобразуем матрицу событий в список, необходимый для использования в процедурах прорисовки:

```
kk:=coerce(x,DOM_LIST)
```

```
[[c7, c8], [c6, c7], [c4, c6], [c1, c4]]
```

11. Осуществим окончательный вывод графического представления решения задачи сетевого планирования (рис. 4):

```
G := Graph(Vert, Ed, EdgeWeights = EdW, EdgeDescriptions = EdD, Directed):
plot(Graph::plotCircleGraph(G,EdgeColor = RGB::Green, SpecialEdges = kk,
SpecialEdgeColor = RGB::Red), Footer="Критический путь")
```

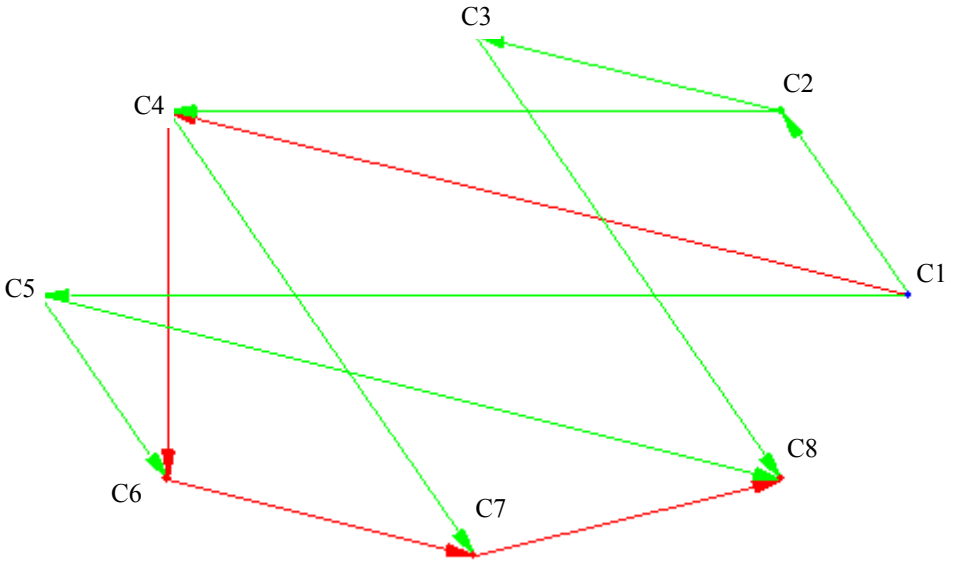

Рис. 4. Критический путь

Предлагаемый алгоритм дает возможность сформировать отдельную библиотечную процедуру, позволяющую решать задачи сетевого планирования в разрезе отыскания критического пути. Дальнейшим этапом развития алгоритма является получение возможности оптимизации производственной программы по временным и стоимостным параметрам [6].

Программная реализация данного алгоритма внедрена и используется в учебном процессе на экономическом факультете Витебского государственного технологического университета в курсах: экономико-математические методы и модели, компьютерные информационные технологии.

ЛИТЕРАТУРА

1. Абчук В.А. Экономико-математические методы. Элементарная математика и логика. Методы исследования операций. – СПб.: Союз, 1999. – 320 с.
2. Похабов В.И. Экономико-математические методы и модели: Практикум. – Мн.: БНТУ, 2003. – 130 с.
3. Бажин И.И. Информационные системы менеджмента. – М.: ГУ-ВШЭ, 2000. – 688 с.
4. Юферева О.Д. Экономико-математические методы. – Мн.: БГЭУ, 2002. – 56 с.
5. Официальный сайт MuPAD <http://www.sciface.com>
6. Костевич Л.С. Информационные технологии оптимальных решений и реинжиниринг в повышении эффективности менеджмента. – Мн.: БГЭУ, 2000. – 108 с.