韓國數學敎育學會誌 시리즈 E
<數學敎育 論文集>
제 17집, 2003. 10. 169-179.

J. Korea Soc. Math. Ed. Ser. E:
Communications of Mathematical Education
Vol. 17, Dec. 2003. 169-179.

알고리즘을 활용한 수학 문제 해결

김 영 미 (신라대학교)
김 향 숙 (인제대학교)
조 용 욱 (신라대학교)

컴퓨터로 문제를 해결함에 있어서 중요한 것은 문제 해결 방법을 찾아내는 것이다. 이렇게 특정 문제를 해결하기 위해 기술한 일련의 명령문을 알고리즘이라고 한다. 본고에서는 학습자의 수학적 창의력을 신장시킬 수 있는 새로운 문제해결의 방법, 즉 알고리즘을 이용하여 해결하는 방법을 여러 예를 통하여 제시하고자 한다.

I. 서 론

제 7차 교육과정은 정보화·세계화로 특징지워지는 21세기를 주도할 수 있는 경쟁력 있는 인간을 효과적으로 양성해내는 것을 목적으로 한다. 21세기 사회는 지식기반 정보화 사회로 특징지워지며, 이에 적합한 교육은 단순 기능인의 양성보다는 자기 주도적으로 지적가치를 추구할 수 있는 자율적이고 창의적인 인간의 육성에 그 중점을 두어야 한다. 이에 대비하기 위한 수학과의 역할은 수학의 기본적인 지식과 기능을 습득하고, 수학의 기본적인 개념, 원리 법칙을 토대로 탐구하고 예측하여 실생활의 여러 가지 문제를 합리적으로 해결하며, 창의적인 문제 해결력을 배양시키는 것이다. 이전의 수학 교육 과정에서도 문제 해결력의 신장은 교육과정의 중요한 목표로 추구되어 왔으나, 제7차 교육과정에서는 문제 해결력과 더불어 창의적 사고력, 논리적 사고력, 비판적 사고력, 문제 해결 능력, 추론 능력, 의사소통 등 제반 고등 사고 능력의 신장을 도모하고 있으며 수학에 대한 자신감과 긍정적인 태도, 수학과 인접학문과의 관련성 및 수학의 유용성 인식과 같이 정의적인 목적을 추구하고 있다 (황혜정 외 5인, 2003). 특히, 제 7차 교육과정에서는 수학적 개념을 이해하는데 도움을 줄 수 있다면 컴퓨터와 계산기의 사용을 적극 권장하고 있다. 컴퓨터를 활용하면 수학적 사실의 직관적 이해와 학습자 스스로의 수학적 탐구에 도움을 줄 수 있기 때문에 수학문제 해결에 있어서 창의성 개발에 많은 도움을 줄 수 있다는 것은 수학적 창의력에 영향을 주는 요인 중에서 환경요인에 대한 중요성을 언급한 다음의 연구에서도 잘 알 수 있다. 교과서, 칠판, 분필만으로 지식을 전달하던 환 경 내에서 학생들의 수학적 창의성을 개발하기란 불가능하다. 학생들이 사고하고 판단하고 스스로 지식을 구성해 갈 수 있도록 자극을 줄 수 있는 환경을 구성해야 한다. 학습내용에 적절한 좌석의 의도적 배치, 개인용 컴퓨터 및 다양한 소프트웨어의 제공, OHP와 실물 화상기 등 다양한 학습기자재의

구비, 충분하고 다양한 학습자료의 비치, 추상적인 개념을 구체화 할 수 있는 모형 등이 풍부하게 구비된 수학교실의 형성이 요청된다 (이대현 외 1인, 1998).

창의성에 관한 수학적 개념은 다음과 같다. 수학적 문제 상황에서 기존의 지식과 경험 등을 바탕으로 정형화된 틀을 벗어나 주어진 문제를 다양한 방식으로 분석하여, 문제의 요소들이나 수학적 아이디어 등을 새로운 방식으로 결합하여 결과를 얻는 것에 관련, 우선 기존 개념의 올바른 도입에 중점을 두어야 하며, 어떤 새로운 결과물을 창출하는 것과도 관련되어 있지만 문제의 새로운 분석, 새로운 접근, 새로운 방식에 보다 밀접하게 관련되어 있다 (김부윤 외 2인, 1999).

수학적 창의력은 다음과 같은 4가지 하위능력을 가진다. 유창성(fluency)은 문제 상황에 유의미한 답으로서 여러 가지 반응 및 아이디어를 낼 수 있는 능력을, 유연성(flexibility)은 서로 다른 범주의 반응 및 아이디어를 낼 수 있는 능력을, 독창성(originality)은 다른 사람과는 다른 참신하며 질적으로도 수준 높은 반응 및 아이디어를 낼 수 있는 능력, 마지막으로 정교성(accuracy)은 산출한 반응 및 아이디어를 보다 구체화하고 세밀하게 다듬을 수 있는 능력을 말한다. 또한 NCTM(1989)의 Standards에서도 21세기를 대비하여 학생들에게 다양한 문제해결 방법에 익숙해 질 수 있도록 확산적이고 창의적인 아이디어를 위한 도전적 과제를 제공해야 하며 한가지 문제를 다양한 방법으로 풀도록 해야한다고 강조하고 있다. 이는 수학적 창의력의 하위능력인 유연성, 독창성, 유창성에 관련되므로 수학적 창의력을 강조 하는 것이라 해석할 수 있다 (조한혁 외 2인, 2002).

따라서 본고는 수학적 문제 해결의 새로운 접근방식 '알고리즘을 활용하여 컴퓨터로 문제를 해결'을 이용하여 학습자의 수학적 창의력을 신장시키는데 그 목적을 두었다.

II. 본 론

컴퓨터로 문제 해결하는데 있어서 첫 번째 가장 중요한 것은 문제 해결 방법을 찾아내는 것이다. 일단 문제 해결 방법을 개발만 하면 이것을 컴퓨터가 이해할 수 있는 특정언어 예를 들어 C나 자바와 같은 프로그래밍 언어로 변환하는 것은 그리 어려운 일이 아니다. 따라서 컴퓨터로 문제를 해결하려 할 때는 컴퓨터 프로그래밍 언어는 잠시 잊어버리고 해결 방법을 정형화하고 그 내용을 명령문 형태로 정확히 기술하는 데에 집중하는 것이 중요하다. 이렇게 특정 문제를 해결하기 위해 기술한 일련의 명령문을 알고리즘(Algorithm)이라고 한다. 컴퓨터로 실제 문제를 풀기 위해서는 결국 이 알고리즘을 컴퓨터가 이해하고 실행할 수 있는 특정 프로그래밍 언어로 표현해야 되는데 이것이 프로그램이다.

◆ 알고리즘을 활용하여 수학적 문제 해결의 과정을 4단계로 구분하여 구성하였다. 그 단계는 다음과 같다.

1) 문제의 이해 단계

문제를 이해하는 단계로 문제에서 구하려는 것과 주어진 것을 알고, 용어의 뜻을 파악하며, 문제를 분석하는 단계이다. 문제에서 구하고자 하는 것은 무엇이며 주어진 조건들은 어떤 관계를 가지고 있는지를 확인한다.

2) 문제를 해결하기 위한 계획을 작성하는 단계

문제에서 주어진 것과 구하려는 것 사이의 관계를 파악하여 여러 가지 문제해결 전략을 세운다.

3) 2)에서 작성한 계획에 따라서 계획을 실행하는 단계

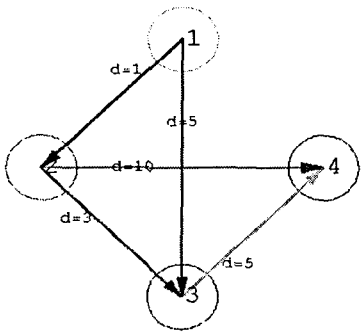
문제의 해결과정을 알고리즘으로 작성한다.

4) 반성

문제를 해결하는 과정을 처음부터 검토해보고, 알고리즘 작성에서의 오류를 수정한다. 또한 다른 알고리즘으로 위의 문제를 해결할 수는 없는지를 알아보고, 혹시 다른 방법이 있으면 어느 방법이 나은지 검토해 본다.

◆ 실전문제 1

다음과 같이 주어진 그래프에서 정점 1에서 정점4까지의 최단 거리를 구하는 간단한 문제를 위에서 제시한 4가지 단계에 따라서 해결해 보자.



주어진 경로

1) 문제의 이해 단계

주어진 그래프에서 정점 1에서 출발하여 정점 4까지 갈 수 있는 모든 경로를 생각하여 가장 짧은 경로를 찾고자 한다. 두 정점 사이의 거리는 d로 나타내고 있으며, 실제 거리의 비와는 다르게 그려져 있다.

2) 문제 해결을 위한 계획의 작성

주어진 문제를 해결하기에는 여러 방법이 있겠지만 여기서는 정점 i에서 j로의 변의 길이가 모두 음이 아닌 즉 모든 $d \geq 0$일 때 유용한 방법으로 잘 알려진 Dijkstra 알고리즘을 활용하여 해결하고자 한다. 이 방법을 따르면 미리 정해진 정점 i로부터 다른 모든 정점에 이르는 최단 경로가 구해진다.

<Dijkstra 알고리즘>

(1) 출발하는 정점의 영구적인 이름을 0으로 부여한다. 그 이외의 다른 모든 정점에는 일시적인 이름 ∞를 부여한다. 모든 정점에 영구적인 이름이 붙여질 때까지 아래의 단계 (2), (3)을 반복한다.

(2) 출발점을 시점으로 갖는 모든 변에 대하여 도착점들이 영구적인 이름을 갖지 않으면서, 그 도착점의 일시적인 이름이 출발점의 영구적인 이름과 거리의 합보다 크면 이 도착점의 일시적인 이름이 출발점의 일시적인 이름과 거리의 합으로 바뀌게 된다.

(3) 일시적인 이름 전체 중에서 최소값을 갖는 것을 하나 택해서 그 정점을 다시 시점으로 한다. 이때 그 시점의 일시적인 이름을 영구적인 이름으로 변환한다.

위 알고리즘을 활용하면 주어진 출발점과 종점사이의 모든 최단 경로를 구해낼 수 있다.

3) 계획의 실행

본고에서는 2단계에서 제시한 알고리즘을 메스메티카(Wolfram, 2000)에 활용하여 실행하는 프로그램을 작성하여 문제를 해결하였다. 각 단계별로 메스메티카 프로그램과 그 실행결과를 구현하였다.

① 정점의 수를 입력한다.

```
n=Input["vertex number="]
```

4

② 각 정점의 위치를 표시한다.

```
<<Graphics`Arrow`
 v[i_]:={n Cos[(2Pi/n)*i],n Sin[(2Pi/n)*i]}
 c[i_]:=Graphics[{Hue[1/20*n i],Thickness[0.003],Circle[{n Cos[(2Pi/n)*i],n
Sin[(2Pi/n)*i]},1]},AspectRatio->Automatic,
            PlotRange->{{-n-2,n+2},{-n-2,n+2}}];
      vertex=Show[Table[c[i],{i,1,n}],AspectRatio->Automatic,
            DisplayFunction->Identity];
      draw[i_,j_]:=Graphics[{Hue[1/20*i],
     Thickness[0.005],Arrow[{v[i][[1]],v[i][[2]]},{v[j][[1]],v[j][[2]]}],Hue[0.5]}];
```

③ 각 정점의 경로를 각각 직접 입력한다.

```
d[i_,k_]:=Input[{i,j}]
Table[Table[dis[i, k]=d[i, k], {k,1,n}], {i,1,n}];
```

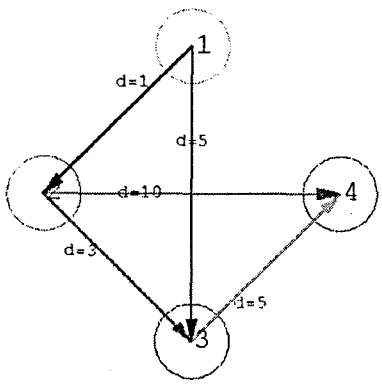
④ 출발점과 도착점 사이의 거리를 나타낸다.

```
TableForm[Table[Table[dis[i,k],{k,1,n}],{i,1,n}],
TableHeadings->{Table[i,{i,1,n}],Table[i,{i,1,n}]}]
```

출발점 \ 도착점	1	2	3	4
1	0	1	5	0
2	0	0	3	10
3	0	0	0	5
4	0	0	0	0

⑤ 전체 경로를 다시 확인해 보자.

```
Show[vertex, Table[Table[If[dis[i, k]  0,
    draw[n, n],draw[i, k]], {k,1,n}], {i,1,n}],
    Epilog->{Table[Text[StyleForm[i, FontColor->Hue[0.7],
    FontSize->18], {n*Cos[(2Pi/n)*i], n*Sin[(2Pi/n)*i]},{-1,0}],{i,1,n}],
  Table[Table[If[dis[i,k]!=0, Text[StyleForm["d="<>ToString[dis[i, k]],
  FontColor->Hue[0.9],FontSize->12],
    {(1.5v[i][[1]]+0.8v[k][[1]])/2, (1.5v[i][[2]]+0.8v[k][[2]])/2}], {}],
    {k,1,n}],{i,1,n}]}, DisplayFunction->$DisplayFunction];
```



⑥ 처음 출발점을 입력한다.

```
s1=Input["start number="]
```

1

⑦ 최종 도착점을 입력한다.

```
f1=Input["final number="]
```

4

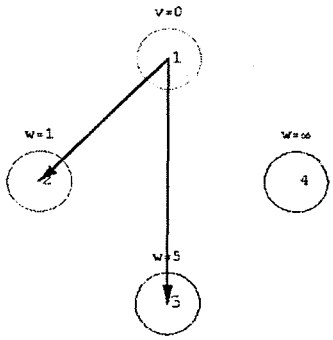
⑧ 각 단계를 dijkstra알고리즘에 의해 실행한다. 내용의 효율성과 정확한 전달을 위하여 이하의 프로그램[1]은 생략하고 그 실행결과만 나타내었다.

<1단계 실행프로그램>

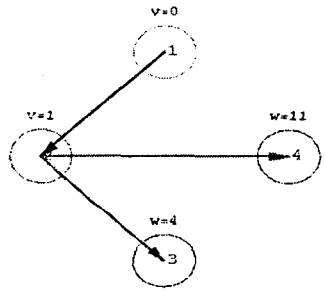
```
next[1]=s1;
Module[{},step1=Table[If[dis[s1,k]!= 0, If[k==f1,draw[s1,k],{}],{}]
      ,{k,1,n}];Table[w[k]=If[s1==k,0, If[dis[s1,k]  0, "∞", dis[s1,k]]],{k,1,n}];
     Show[vertex, draw[s1, next[1]], step1,
      Epilog->{Table[Text[StyleForm[i,FontColor->Hue[0.9],FontSize->14],
          {n*Cos[(2Pi/n)*i], n*Sin[(2Pi/n)*i]},{-1,0}],{i,1,n}],
       Table[If[k  s1,Text[StyleForm["v="<>ToString[w[k]],
       FontColor->Hue[0.8],FontSize->12],{v[k][[1]],v[k][[2]]+1.5}],
       Text[StyleForm["w="<>ToString[w[k]],FontColor->Hue[0.8],
           FontSize->12],{v[k][[1]],v[k][[2]]+1.5}]],{k,1,n}]},
           DisplayFunction->$DisplayFunction];
      Table[w[k]=If[s1==k,0, If[dis[s1,k]  0, ∞ , dis[s1,k]]],{k,1,n}]];
```

1) 본 프로그램에서는 영구적인 이름을 v로 일시적인 이름을 w로 구현하였다.

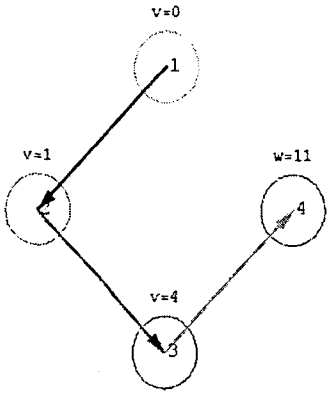
◆ 실행결과



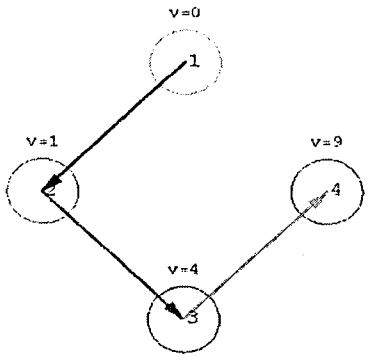
1단계- 처음 출발 정점 1에서 갈 수 있는 모든 경로



2단계- 두 정점 중에서 최소값을 가지는 정점에 영구적 이름(v)을 부여하고, 그 정점이 다시 시점이 되어 모든 경로를 나타낸다.



3단계- 2단계를 반복한다.



4단계- 모든 정점이 영구적인 이름(v)을 부여받을 때까지 반복한다.

⑨ 최단거리를 확인한다.

```
If[next[4]==f1, Print[StyleForm["최단거리는="<>ToString[w4[f1]],
    FontColor->Hue[0.8], FontSize->12]],next[5]=Input["minimum"]];
```

최단거리는 = 9

⑩ 결론

4단계에서 확인할 수 있듯이 정점 1에서 정점 4까지의 최단 경로는 (1, 2, 3, 4)이고, 최단거리는 최종 도착점인 정점 4의 영구적인 이름으로 9임을 알 수 있다.

4) 반성

위 예제에서 정점 사이의 거리 d를 거리뿐만 아니라, 두 정점을 가기 위한 경비의 문제로도 생각하면 같은 알고리즘의 구현으로 최소 경비의 문제를 해결할 수 있다. 즉 2단계에서 소개한 알고리즘은 다양한 실생활 문제에 적용할 수 있는 알고리즘이라 알 수 있다. 하지만 이 알고리즘만이 유일한 알고리즘은 아니다. 지금까지 알려진 최단 경로에 적용되는 알고리즘은 아주 많으며 다른 알고리즘으로 구현했을 때 더 좋은 결과를 얻을 수 있다면 충분히 문제해결의 중요한 전략이 될 것이다.

◆ 실전문제2

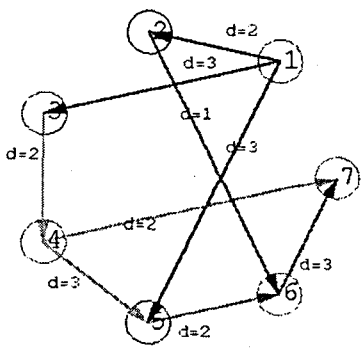
다음 도로망에서 지점 x에서 지점 y에 이르는 최단 경로를 구하여라.

도로망의 경로는 다음과 같다.

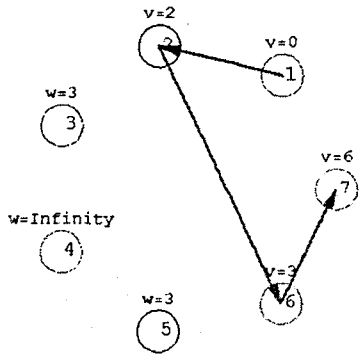
출발 \ 도착	1	2	3	4	5	6	7(y)
1(x)	0	2	3	0	3	0	0
2	0	0	0	0	0	1	0
3	0	0	0	2	0	0	0
4	0	0	0	0	3	0	2
5	0	0	0	0	0	2	0
6	0	0	0	0	0	0	3
7	0	0	0	0	0	0	0

답) 이 문제 역시 위의 프로그램을 적용시켜보면 최단거리가 6임을 알 수 있고 그 경로는 (1, 2, 6, 7)임을 쉽게 확인할 수 있다.

◆ 실행결과



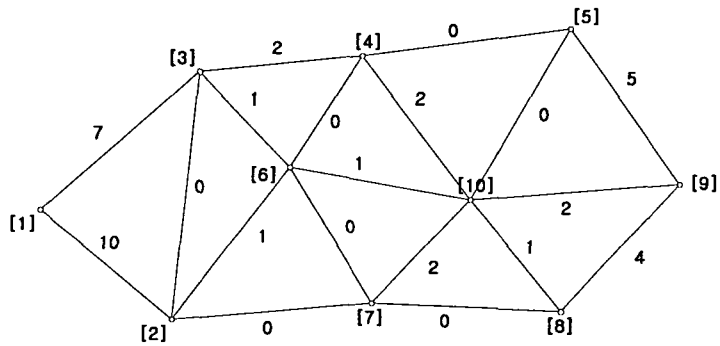
전체 경로



최종 경로

◆ 실전문제3

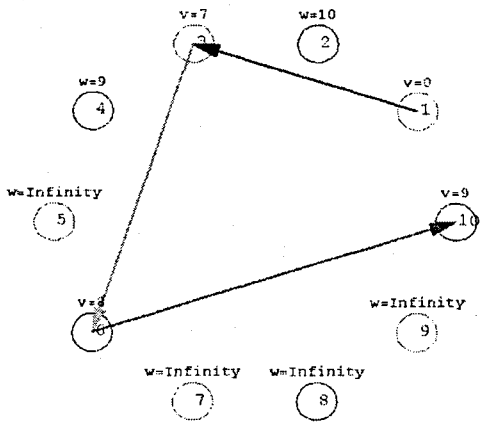
아래의 그래프는 x지점에서 y지점까지($x<y$) 가는 유료 고속도로망을 나타낸 것이고 각 변의 무게는 그 도로를 통과하기 위한 요금이다.



정점 1에서 정점 10까지 이르는 최소 경비를 구하여라.

답) 이 문제 역시 위의 프로그램을 적용시켜보면 최소경비가 9임을 알 수 있고 그 경로는 (1, 3, 6, 10)임을 쉽게 확인할 수 있다.

◆ 실행결과



최종 경로

III. 결론 및 제언

문제 해결에서 가장 중요한 아이디어는 학생의 사고로 탐색하되, 그 이외의 문제해결의 수단이 되는 복잡한 공학적 도구의 다양한 기능을 이용하게 하는 것이다. 이런 방식으로 문제해결에 공학적 도구를 도입하면, 학교 수학에서 학생들의 현실과의 관련성으로 충만한 문제들을 다룰 수 있게 된다. 이는 결국 학생들이 주위의 여러 가지 현상을 수학적 안목으로 파악하도록 한다는 수학교육의 본질적인 목적을 추구하는 데도 일조를 할 수 있을 것이다(황혜정 외 5인, 2003).

제 7차 교육과정에서 중시하고 있는 부분 중의 하나가 바로 논리적, 창의력을 신장하는 것이다. 알고리즘이라는 것은 몇 단계의 절차를 거쳐 답을 구해내고자 하는 것으로 만약 논리성에 어긋나거나 잘못된 연산을 통하게 되면 정확한 문제 해결을 할 수 없게 된다. 그러므로 문제해결 방법에 있어서 그 전략으로 알고리즘을 활용하는 것은 논리력 향상에 많은 도움을 주게 된다고 생각한다. 다만 단순 계산만을 반복하는 알고리즘이라면 그러한 효과를 기대하기가 함들 수 있지만 문제해결을 위한 알고리즘의 구현으로 컴퓨터에 직접 프로그램을 하여 실행해 봄으로써 새로운 문제 해결 방법으로 제시될 수 있으므로 학습자의 창의력신장에도 많은 도움을 주리라 생각한다.

본고에서는 학습자의 창의성과 논리성을 동시에 신장시키고자하는 방법으로 알고리즘을 활용하여 직접 컴퓨터 프로그램으로 실행하는 문제접근 방법을 제시하여 보았다. 아직 많은 부분에 미흡하리라 생각하지만 끊임없이 변화해 가면서 수정, 보완해 갈 것이다.

참 고 문 헌

김부윤 · 이지성 · 조재호 (1999). 중등수학교육에 있어서 창의성 교육의 실제에 관한 연구, 대한수학교육학회 춘계 수학교육학 연구 논문 발표 대회.

김안현 · 김향숙 · 류재칠 · 신준용 · 이강래 · 표용수 (2000). 수학에서의 Mathematica활용. 서울: 경문사.

이대현 · 박배훈 (1998). 수학적 창의력에 대한 소고, 대한수학교육학회 논문집 2, pp.679-690, 서울: 대한수학교육학회.

- 조한혁 · 안준화 · 우혜영(2002), 컴퓨터를 통한 창의력 수학 프로그램 개발, 한국수학교육학회지 시리즈 E <수학교육 논문집> 13, pp. 625-639, 서울: 한국수학교육학회.
- 황석근 · 이재돈 · 김익표 (2001). 이산 수학, 블랙박스.
- 황혜정 · 나기수 · 최승현 · 박경미 · 임재훈 · 서동엽 (2003). 수학교육학 신론.
- NCTM (1989). *Curriculum and Evaluation Standards for school Mathematics*, Reston, VA: The NAtional Council of Teachers of Mathematics, Inc.
- Wolfram, S. (2000). *Mathematica(4th, ed.)*, Cambridge Univ. Press.