페트리 네트 기반 멀티미디어 시나리오 작성기 설계 및 재생기 구현

임 재걸, 이 계영
(동국대학교 전자계산학과)

A Design of a Petri-Net-Based Multimedia Scenario Authoring Tool and Implementation of a Scenario Player

Jaegeol Yim, and Gyeyoung Lee
Dept. of Computer Science, Dongguk University at Kyung-Ju

요 약

멀티미디어 시나리오를 표현할 수 있는 페트리넷을 제안하고, 멀티미디어 시나리오 작성기 제작에 이 페트리넷을 이용하는 방법을 제안한다. 또한 제안된 페트리넷으로 표현된 멀티미디어 시나리오를 재생하여 주는 시스템 구현을 소개한다. 구현된 재생기는 처음부터 끝까지 차례로 출력하는 것 뿐만 아니라 '정지', '역방향 재생', '빠른 재생', '빠른 역방향 재생' 등의 기능도 갖고 있다.

1. 서 론

전문 프로그래머가 아닌 사람들이 쉽고 빠르게 멀티미디어 프로그램을 작성할 수 있도록 환경을 제공하여 주는 프로그램인 멀티미디어 저작 도구로 MINOS, Multos, Toolbook, 새빛[1] 등을 들 수 있다. 기존의 방식은 먼저 화면 단위로 흐름도를 작성한 다음 각 화면의 구성 요소를 배치하는 방법으로 시나리오를 작성한다. 이러한 화면 단위의 작성 방법은 여러 화면에 걸쳐 화면 진행에 독립적으로 출력되는 데이터를 표현하는 데 많은 어려움이 있다.

본 논문에서는 페트리네트[2]를 이용한 시나리오 작성기 설계를 제안하는데, 이 방법에서는 흐름도의 구성 단위가 윈도우이며, 따라서 화면 단위 방법이 갖는 단점이 이 방법에는 제거된다. 제안된 작성기는 멀티미디어 시나리오를 페트리넷으로 표현하게된다. 이러한 페트리넷을 입력으로 받아 시나리오 대로 멀티미디어를 출력시켜 주는 재생기의 구현도 소개한다.

본 재생기는 START 플레이스에서 END 플레이스로의 순차적인 흐름 뿐 만 아니라, 흐름의 정지, 역 방향 재생 및 빠른 재생 등도 가능하다.

2. 멀티미디어 페트리네트

본 논문에서는 멀티미디어 프로그램의 시나리오를 모델링하기 위하여 기존의 페트리네트를 다음과 같이 변형하여 사용한다. 본 논문에서 제안하는 페트리네트는 기존의 OCPN(Object Composition Petri Net)[3]에 비동기 전략을 추가한 것으로, 이를 MPN(Multimedia Petri Net)이라고 한다.

2.1 MPN의 정의

[정의 1] MPN = (P, MT, POS_s, W_POS, RECT, PATH, T, F_{syn}, E, FMT, FPOS_s, FW_Pos, FRECT, FPATH),

P = {p_1, p_2, ..., p_n} : place의 집합으로 그래프에서는 원으로 표현됨.

MT = {MT_start, MT_end, MT_bitmap, MT_avi, MT_wave, MT_midi, MT_mpeg, τ} : 데이터 유형의 집합으로 τ는 시간 지연.

POS_s = {(X_pos, Y_pos)} : X_pos와 Y_pos는 음

이 아닌 정수이며, 시나리오 작성시의 스크린 상의 위치임.

W_POS = {(X_pos, Y_pos)} : 윈도우의 위치.

RECT = {(Width, Height)} : Width와 Height는 음이 아닌 정수로, 윈도우의 크기.

PATH: 화일 이름을 포함한 경로.

$T = \{t_1, t_2, \ldots, t_n\}$: transition의 집합으로 그래프에서 사각형으로 표현됨.

$F_{syn} : T \rightarrow P^*$, t_i는 $F_{syn}(t_i)$에 나열된 place들을 격발에서 무시하며, 따라서 이들을 Exclude 리스트라 부름.

$E \subseteq \{T \times P\} \cup \{P \times T\}$: 간선의 집합으로, 토큰의 흐름을 나타냄.

FMT, FPOS_s, FW_Pos, FRECT, FPATH : 각각의 place에 연관된 MT, POS_s, W_Pos, RECT, PATH를 찾아 주는 함수임.

본 논문에서 정의한 MPN 모델에서는 place를 멀티미디어 데이터로 나타내고, transition은 멀티미디어 출력의 천이를 나타낸다. 그리고, Place와 transition의 합집합인 정점 V에서, (v_i, v_j)가 간선일 때 v_i를 v_j의 입력 정점이라 하고, v_j를 v_i의 출력 정점이라 한다. 어떤 transition t_i는 입력 정점(place)에 연합된 데이터의 출력이 모두 끝날 때까지 출력 정점에 연합된 데이터의 출력을 지연시키고 있다가, 입력 정점의 데이터 출력이 끝났을 때 비로소 출력 정점에 연합된 데이터의 출력을 동시에 시작한다. 이러한 경우에는 $F_{sync}(t_i)$의 원소인 입력 place는 무시하게 된다.

즉, 이들은 출력의 완료 유무에 관계없이 입력 장소의 데이터 출력이 완료되면 출력 장소의 데이터 출력을 시작한다. 이와 같은 데이터 출력의 천이는 토큰으로 나타내게 되는데, 어떤 place에 연합된 데이터가 출력 중이면 그 place에 토큰을 놓아 출력 중임을 나타낸다. 그리고, 페트리네트 분야에서는 트랜지션의 입력 place에 있는 토큰들이 출력 장소로 옮겨 가는 것을 격발(firing)이라고 한다.

2.2 MPN의 변천 격발 규칙

MPN에서는 place에 토큰이 놓이면 해당 데이터의 출력을 시작하고, 변천의 모든 입력 장소에 연합된 데이터의 출력이 종료하게 되면 변천을 격발함으로써 입력 장소의 토큰을 제거하여 출력 장소에 옮겨 놓은 후, 새로운 데이터의 출력을 시작시킨다. 그러므로, 시나리오의 흐름은 $FMT(p_i)$가 MT_start인 place p_i에 한 개의 토큰을 놓음으로써 시작하게 된다. 다음은 출력의 천이를 제어하는 MPN의 변천 격발 규칙을 보인 것이다.

1) MPN은 초기에 $FMT(p_i)$가 MT_start인 place p_i에 한 개의 토큰을 갖는다.

2) place p_i에 토큰 tok_k가 놓이면 $FW_Pos(p_i)$에 $FRECT(p_i)$ 크기의 윈도우를 띄우고, 여기에 $FPATH(p_i)$에 지정된 데이터를 출력시킨다. 출력이 완료되면 p_i의 out going 간선 (p_i, t_j)을 enable 시킨다. 이때 p_i가 $F_{sync}(t_j)$의 원소이면 p_i의 데이터를 출력시킴과 동시에 (p_i, t_j)를 즉시 enable 시키고, tok_k에 (p_i, t_j)라는 레이블을 부친다.

3) 모든 입력 간선이 enabled된 변천 t_i는 즉시 격발한다. 격발은 t_i의 입력 장소 p_j의 토큰과 (p_j, t_i)라는 레이블을 갖는 모든 토큰을 제거하고, 출력 장소에 한 개씩의 토큰을 놓는다.

4) $FMT(p_i)$가 MT_end인 place p_i에 토큰이 놓일 때 격발을 종료한다.

그림1은 '천재들의 과학백과'라는 학습용 CD-ROM 타이틀에서 중력을 설명하는 부분인데 첫 화면에는 두 개의 이미지 파일(중력.bmp, 스카이다이버.bmp)과 한 개의 오디오 파일(중력.wav)이 동시에 출력되고, 중력.wav의 출력이 종료되면 비디오 파일(무중력.avi)이 출력된다. 이미지 파일들은 비디오 파일이 출력되는 동안에도 계속 화면에 남아 있는다. 이러한 출력 데이터들은 그림1의 원들의 레이블로 표기되어 있음을 알 수 있다.

두 번째 화면과 세 번째 화면은 각각 두 개의 이미지 파일과 한 개의 오디오 파일로 구성된다. 그림1에는 나타나 있지 않지만 각 원에는 레이블로 지정된 데이터가 출력될 화면상의 위치, 크기 등의 데이터도 연합되어 있다. 이 시나리오 예제의 격발행렬은 [표.1]과 같다. 여기서 플레이스(원)에서 트랜지션(막대)으로의 유향간선은 1로, 그 반대는 2로 표현된다.

t \ p	0	1	2	3	4	5	6	7	8	9	10	11
0	1	2	2	2								
1				1	2							
2		1	1		1	2	2	2				
3						1	1	1	2	2	2	
4									1	1	1	2

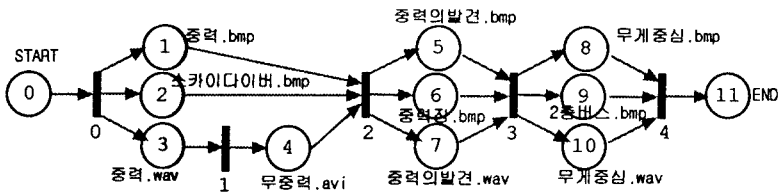
[표.1] 예제의 격발 행렬

[Table.1] Incidence Matrix of Example

3. 페트리 네트로 표현된 멀티미디어 시나리오 재생기 구성과 작동

시나리오 재생기는 흐름을 담당하는 부분과 플레이스를 해당 미디어로 프리젠테이션 하기 위한 부분들로 양분할 수 있다.

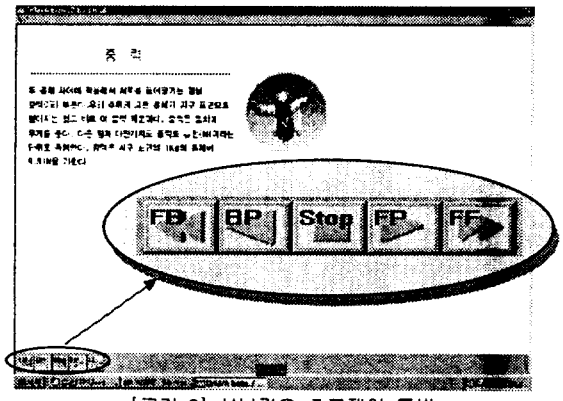
먼저 흐름을 담당하는 부분은 CDMAEngine 이라는 클래스와 Run이라는 쓰레드 함수가 주축을 이룬다. CDMAEngine 클래스는 격발행렬(Incidence Matrix)을 나타내는 클래스(CIMatrix)의 인스턴스(m_cimx)와 트랜지션과 플레이스를 관리하는 클래스(CTPManager)의 인스턴스(m_ctpm)를 포함하고 있다. 시나리오 작성시 사용자로부터 획득된 정보(좌표, 파일명, 타입 등)는 m_ctpm에 적절히 관리되고, 플레이스와 트랜지션 사이의 간선으로 표현된 정보는 격발 행렬의 인스턴스(m_cimx)에 [표.1]과 같은 형태로 저장된다. 격발 행렬의 흐름은 쓰레드 함수 Run()에서 실현 한다. 이 함수는 invoke될 때 플레이스 ID를 하나 전달 받는다. 그러면 전달받은 ID의 플레이스를 먼저 수행하고 수행이 끝나면 이 플레이스의 out put transiton을 찾아 활성화 되었는지 본다. 즉, 트랜지션이 다른 플레이스로부터도 활성화를 기다리고 있다면 현재 수행중인 Run() 쓰레드는 종료한다. 기다리는 플레이스가 없다면, 즉, 트랜지션이 기다리는 플레이스가 하나 뿐이거나, 마지막 플레이스인 경우, 트랜지션은 활성화된다. 트랜지션의 격발이 생성하는 토큰 개수만큼의 쓰레드를 생성하여 각각에 대해 토큰이 놓일 장소의 플레이스 ID를 인수로 하여 Run() 루틴을 병행으로 재귀 호출한다. Run() 함수의 골격은 다음과 같다.



[그림.1] '중력'을 설명하는 간단한 페트리 네트 시나리오 예제
[Fig.1] Simple Petri Net-based Multimedia Scenario Example that Explains 'Gravity'

```
Run(플레이스 ID)
begin
   플레이스실행(플레이스ID);

   토큰을발사할트랜지션을찾아라();

   if (트랜지션이 없다)
      리턴;

   if (도착하지않은토큰이있다)
      리턴;

   병행처리
   begin

      Run(트랜지션이 활성화할 플레이스들);
   end

   리턴;
end
```

미디어를 제어하는 부분은 매체당 하나의 클래스를 가지고 있다. 이 클래스들은 플레이스를 실행할 때, 플레이스의 정보로 먼저 인스턴스화 되어 있는 이 클래스 객체를 세팅하고 실행한다.



[그림.2] 시나리오 흐름제어 툴바
[Fig.2] Toolbar for Controlling The Flow of a Scenario

4. 다양한 재생 방법을 위한 구현 방법의 변형

본 재생기에는 일반인적인 카세트 플레이어에 제공되는 5가지의 제어 방법이 구현되어 있다. 구현된 제어 방법은 [그림.2]에 보이는 바와 같이 FB(Fast Backward), BP(Backward Play), ... 등 다섯 가지이다. 이를 위하여 entry 3을 추가한다. 예를 들면, 어떤 플레이스 p_i에 해당하는 데이터 출력이 종료하고 그 플레이스의 out put transition이 t_j 일 때, 격발행렬 IMatrix[j,i]에 3을 넣는다.

본 연구에서 5 가지 제어 방법은 각각 다음과 같이 정의된다. 기본 흐름인 '재생(Play)'을 기준으로 '역방향 재생(Backward Play)'은 '재생'과 흐름의 방향만 반대이고 나머지는 같은 것으로 정의한다. '정지(Stop)'는 트랜지션 상에서만 이루어 지는 것으로 정

의 한다. 즉 수행 중인 플레이스를 강제로 종료시키고 정지할 수는 없다는 말이다. 이것은 이미지 플레이스의 경우 시간성을 가지고 있지 않고, 소리나 동영상의 경우는 시간성을 가지므로 수행 완료를 플레이스(미디어) 단위로 통일하여 흐름을 제어하기 위해서이다. '빠른 재생(Fast Play)'이나 '빠른 역방향 재생(Fast Backward Play)'은 시간성을 가지는 플레이스들을 실행하지 않고 skip하여 흐름을 빠르게 하는 것으로 정의한다.

이렇게 정의된 5 가지의 재생 방법을 나타내기 위해, 재생 방향을 나타내는 플래그인 bForward와 빠른 수행 여부를 나타내는 플래그인 bFast, 그리고 정지 상태를 나타내는 플래그인 bStop 등 3 개의 플래그를 둔다. 이들 플래그로 5 가지 재생 모드를 나타내자면 [표. 2]와 같다.

재생모드 / 플래그	◀◀	◀	■	▶	▶▶
bForward	F	F	F or T	T	T
bFast	T	F	F	F	T
bStop	F	F	T	F	F

[표.2] 재생 모드 플래그와 재생 모드
[Fig.2] Flag for Playing Mode and Playing Mode

쓰레드 함수 Run()은 트랜지션이 활성화 되고 해당 플레이스를 활성화 시키는 과정에서 플래그들을 체크하도록 변형한다. 예를 들면, 플래그 bStop이 true로 세트되어 있으면, 현재 활성화된 트랜지션 ID를 스택에 저장하고, 쓰레드를 종료한다.

플레이스 실행시 미디어 별로 해당 함수를 플레이스 ID로 호출해 주는 함수도 bFast 플래그에 따라 시간성 플레이스를 skip하도록 수정된다. 예를 들면 bFast플래그가 true로 세팅되어 있는 경우는 wave나 avi, mpeg 플레이스는 실행하지 않는다.

5. 구현

본 시스템은 Pentium II - 200MHz, Windows95 시스템에서 Microsoft Visual C++ 5.0, MFC 4.0의 개발 환경에서 구현했다. 흐름 제어 툴바 [그림.2]에서 사용자가 원하는 재생 모드가 선택되었을 때 모드에 따라 플래그가 세팅되는 루틴은 [리스트.1]과 같다.

```
OnStop() {
    bFast = false;
    bStop = true;
}

/* Forward Play */
OnPlay() {
    bForward = true;
    bFast = false;
    if ( bStop ) {
        bStop = false;
        Resume();
    }
}

/* Fast Forward Play */
OnFastPlay() {
    bForward = true;
    bFast = true;

    if ( bStop ) {
        bStop = false;
        Resume();
    }
}

/* Backward Play */
OnBackPlay() {
    bForward = false;
    bFast = false;

    if ( bStop ) {
        bStop = false;
        Resume();
    }
}

/* Fast Backward Play */
OnFastBackPlay() {
    bForward = false;
    bFast = true;
    if ( bStop ) {
        bStop = false;
        Resume();
    }
}
```

[리스트.1] 재생 모드에 따른 플래그 세팅 루틴
[List.1] The routine for Setting Flag of Play Mode

세팅된 플래그에 따라 격발 행렬을 처리하는 함수는 각기 다른 동작을 만들어 낸다. 다시 말해, 방향 플래그(bDirection)와 정지 플래그(bStop)에 따라 다른 '재생'과 '역방향 재생' 및 '정지' 모드를 취한다. 변형된 Run() 함수는 [리스트.2]와 같다.

```
Run(placeID)
begin
   place ← GetPlace(placeID);
   Distribute(place);

   for ( 0 < i < MaxTransition )인 모든 i,
      if ( bForward )
         IMatrix[i,placeID] = TOKEN_GET 인 i를 찾아라;
      else
         IMatrix[i,placeID] = TOKEN_FIRE 인
         i를 찾아라;

   if (찾지못했으면)
      return -1;

   if ( bForward )
```

```
    IMatrix[i,placeID] ← TOKEN_DONE;

  for ( 0 < j < MaxPlace)인 모든 j,
    if ( bForward )
      if ( IMatrix[i,j] = TOKEN_GET )
        return 1;
    else
      if ( IMatrix[i,j] = TOKEN_FIRE )
        return 1;

  /* 트랜지션을 활성화 하기 전에 격발
     행렬의 정보를 되돌린다. */
  for ( 0 < m < MaxPlace)인 모든 m,
    if ( IMatrix[i,m] = TOKEN_DONE )
      if ( bForward )
        IMatrix[i,m] = TOKEN_GET;
      else
        IMatrix[i,m] = TOKEN_FIRE;

      EngImages(); /* 화면잔상을 제거 */

  / * 정지 플래그가 세트 되어 있는 경우 */
  if ( bStop )
    Push(i);
    return 1;

  for ( 0 < k < MaxPlace )인 모든 k,
    if ( nForward )
      if ( IMatrix[i,k] = TOKEN_FIRE )
        /* 병행 수행 */
        Run(k);
    else
      if ( IMatrix[i,k] = TOKEN_GET )
        /* 병행 수행 */
        Run(k);

  return 1;
end.
```

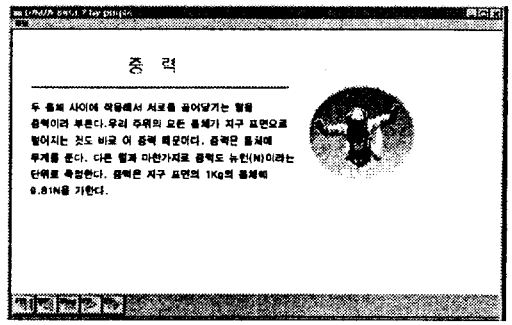
[리스트.2] 방향 플래그에 따라 격발 행렬을 처리하는 쓰레드 함수 Run()

[List.2] Thead Function Run() which manupulates Incidence Matrix according to Direction and Stop Flags

플레이스의 타입(start, end, avi, bmp, mpg, ...)에 따라 처리 함수를 호출해주는 Distribute 함수에서 bFast 플래그를 검사한다. bFast가 true로 설정되어 있으면, wave와 동영상 파일은 호출하지 않고 skip하므로 빠른 (역방향) 재생을 한다.

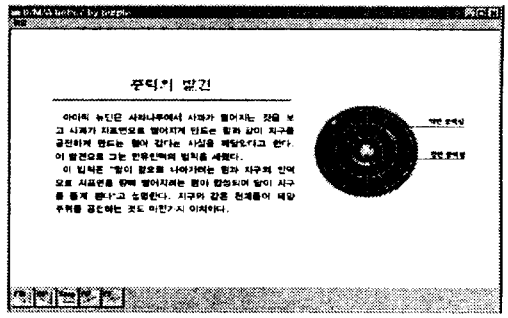
6. 실험

본 논문에서는 [표.1]의 격발 행렬을 같는 [그림.1]의 시나리오를 실험 데이터로 하여 다양한 재생 방법을 실험해 보았다. 프로그램 초기화 후, '재생' 버튼을 누르면, 플레이스 0에서 프로그램이 시작된다. 곧이어 플레이스 1, 2, 3이 병행 실행된다[그림.3]. 이때 툴바의 '빠른 재생버튼'을 누른다. 그러면 플레이스 3(중력.wav)이 끝나자마자, 플레이스 4를 skip하게 되고, 바로 플레이스 5, 6이 실행된다.(플레이스 7 skip) [그림.4]. 이때 다시 역방향 재생을 눌러 재생 모드를 바꾼다. 그러면 플레이스 5, 6의 실행이 끝나고 다시 플레이스 5, 6, 7이 실행된다. 이들의 실행이 끝나면 다시 플레이스 1, 2, 4가 실행되고 플레이스 4가 끝나면 다시 플레이스 3이 실행되고, 세 개의 플레이스 1, 2, 3의 수행이 모두 끝나면 플레이스 0이 마지막을 수행되어 종료한다. 이 경우, 역방향 재생으로 시작 플레이스까지 실행하고 종료된 경우이다.



[그림.3] 시작후, 플레이스 1,2,3이 병행 실행되고 있다.

[Fig.3] After program started, Place 1,2,3 are being run paralle



[그림.4] 플레이스 5, 6이 실행중이다.

[Fig.4] Place 5,6 are being run.

7. 결 론

멀티미디어 시나리오를 표현하기에 알맞은 페트리넷을 제안하고, 이러한 페트리넷으로 표현된 시나리오를 재생하여 주는 재생기를 실제 구현하여 보았다. 기존의 단순한 순차적인 흐름은 실제 멀티미디어 시스템에 적용되기엔 부적합하다. 사용자는 스스로가 그 흐름을 제어하기 원할 때도 있기 때문이다. 본 연구에

서 기존의 시나리오 흐름에 역방향과 정지 및 빠른 재생의 기능을 추가 함으로써 좀 더 실질적인 시나리오 재생기를 구현 및 실험한 결과를 소개 하였다. 본 논문에 제안된 재생 방법의 정의에 의하면, 트랜지션에서만 다른 재생 모드로의 전환이 가능하다. 향후 연구 계획은 플레이스 실행중에 다른 모드로의 전환, 시간 지연 플레이스 도입, 동기화 기법의 다양화 등의 기능을 첨가하는 것이다.

참고문헌

[1] 정성무 외, *멀티미디어 저작 도구 새빛을 배우자*, 교학사, 1996.

[2] T. Murata, *"Petri Nets: Properties, Analysis and Applications,"* Proceedings of the IEEE, Vol 77, No 4, April 1989.

[3] M. Diaz, P. Senac, "Time Stream Petri Nets a Model for Timed Multimedia Information," *Application and Theory of Petri Nets 1994*, Springer-Verlag, 1994, pp. 219-238.