Journal of the Society of Korea Industrial and Systems Engineering
Vol, 31, No. 2, pp.104－111, June 2008.

연관규칙과 순차패턴을 이용한 프로세스 마이닝

정소영 · 권수태†

전주대학교 정보시스템학과

A Process Mining using Association Rule and Sequence Pattern

So Young Chung · Soo Tae Kwon†

Department of Information System, Jeonju University

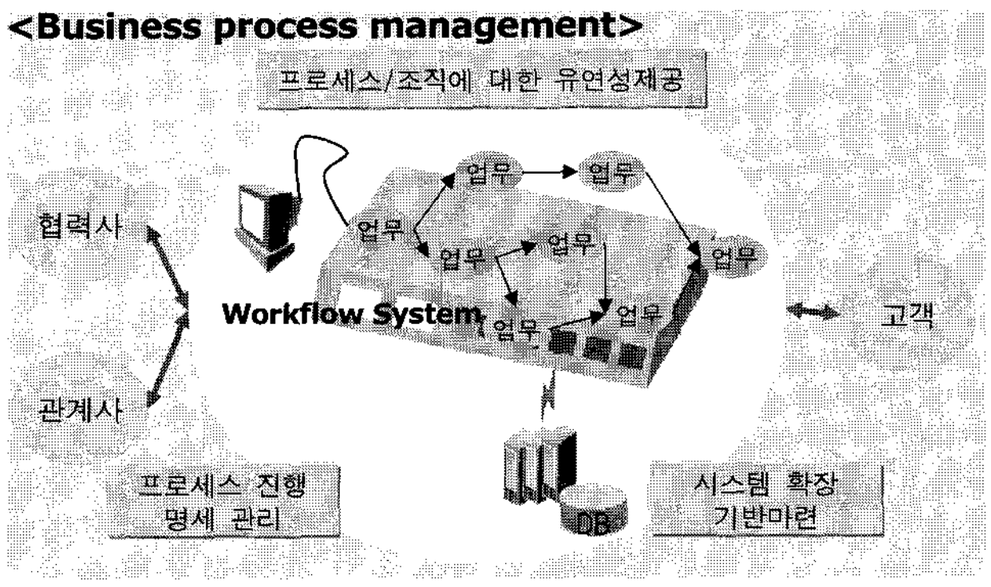
A process mining is considered to support the discovery of business process for unstructured process model, and a process mining algorithm by using the associated rule and sequence pattern of data mining is developed to extract information about processes from event-log, and to discover process of alternative, concurrent and hidden activities. Some numerical examples are presented to show the effectiveness and efficiency of the algorithm.

Keywords : Process Mining, Business Process, Data Mining, Association Rule, Sequence Pattern

1. 서 론

경영환경의 변화가 가속화되면서 많은 기업들은 환경변화에 빠르게 적응하기 위하여 인프라구축, 업무처리 생산성 향상, 기업내부의 역량강화, 경쟁력 확보 등의 노력을 기울이고 있다. 특히, 비즈니스 프로세스의 혁신을 통해 경쟁력을 향상시키기 위해 점점 그 중요성이 높아지고 있는 비즈니스 프로세스 관리(BPM : Business Process Management)에 관심을 가지고 있다.

BPM은 기업 내 혹은 기업과 기업, 고객과 기업 사이에 일어날 수 있는 비즈니스 프로세스를 효율적으로 구축하고 관리하기 위한 것으로, 비즈니스 프로세스를 시스템 차원에서 관리하여 프로세스에 대한 자동화와 생산성 및 효율성의 확대, 프로세스에 대한 지식을 축적하고 분석 · 개선하는 것을 목적으로 한다. 또한 BPM의 핵심기술인 워크플로우 시스템은 업무 자동화와 프로세스 설계, 프로세스 단위구분 등을 위해 <그림 1>과 같이 목적단위 업무간의 흐름을 분석 · 통제 · 관리하여, 정확하고 신속한 업무처리, 효과적인 정보제공과 실시간 업무통제 등을 할 수 있다. 그리고 체계적이고 계속적인 워크플로우 평가와 분석을 통해 비즈니스 프로세스 혁신을 수행할 수 있다.



<그림 1> 비즈니스 프로세스와 워크플로우 시스템

BPM이나 워크플로우 시스템을 통해 비즈니스 프로세스를 효율적으로 수행 · 관리하기 위해서는 비즈니스 프로세스 상의 모든 업무 수행자들과 지속적인 의사소

논문접수일 : 2008년 03월 26일 논문수정일 : 2008년 06월 04일 게재확정일 2008년 06월 10일
† 교신저자 kstfms@jj.ac.kr

통을 통해 비즈니스 프로세스에 대한 지식을 습득하여 업무에 대한 정보를 명세화 시켜야 한다. 하지만, 비즈니스 프로세스가 직관적이거나 표준적인 행동에 기반을 두지 않고, 업무특성에 따라 각각의 정보시스템을 사용하여 각 시스템에서 규정한 형식에 따라 비즈니스 프로세스의 업무수행에 대한 정보를 저장하기 때문에 프로세스 정보를 명세화 하는 것은 쉽지 않다.

따라서 본 연구에서는 기업에서 이용하는 각 정보시스템의 프로세스 수행 정보가 저장되어 있는 트랜잭션 로그(transaction log)를 대상으로, 많은 양의 데이터를 효과적으로 분석하고 유용한 정보를 발견하는 데이터마이닝의 연관규칙과 순차패턴을 적용하여 보다 효율적이고 효과적인 프로세스를 발견하고자 한다.

본 논문의 구성은 다음과 같다. 제 2장에서 프로세스 마이닝과 기존연구에 대해 설명하고, 제 3장에서는 연관규칙과 순차패턴을 이용한 프로세스 마이닝 알고리즘에 대해 기술한다. 제 4장에서는 개발된 알고리즘의 적용사례와 성능평가를 위한 비교실험을 수행하고, 제 5장에서는 본 연구의 결과 및 향후 연구방향을 제시한다.

2. 프로세스 마이닝

프로세스 마이닝은 기업의 비즈니스 프로세스에서 일어나는 업무처리기록을 바탕으로 유용한 정보를 발견(discover)하는 것을 목적으로 하며, 프로세스 마이닝의 결과는 기업의 비즈니스 프로세스 혁신에 활용될 수 있다[2].

```
Case 1
Diractive Description  Event          User          yyyy/mm/dd hh:mm
-------------------------------------------------------------------
                       Start          jvluin@staffw 2002/04/16 11:06
Register complaint     Processed To   jvluin@staffw 2002/04/16 11:16
Register complaint     Released By    jvluin@staffw 2002/04/16 11:26
Evaluate complaint     Processed To   jvluin@staffw 2002/04/16 11:36
Evaluate complaint     Released By    jvluin@staffw 2002/04/16 11:46
                       Terminated                   2002/04/16 11:56
Case 2
Diractive Description  Event          User          yyyy/mm/dd hh:mm
-------------------------------------------------------------------
                       Start          jvluin@staffw 2002/04/16 12:36
Register complaint     Processed To   jvluin@staffw 2002/04/16 12:46
Register complaint     Expired        jvluin@staffw 2002/04/17 13:07
Register complaint     Withdrawn      jvluin@staffw 2002/04/17 13:07
```

<그림 2> Staffware 로그파일 예

비즈니스 프로세스 상의 모든 업무는 기업 내부의 정보시스템인 ERP, CRM, SCM, Workflow 등에서 수행되고, 이들 정보시스템에서는 모든 트랜잭션(transaction)을 <그림 2>와 같이 이벤트 형식으로 프로세스 인스턴스(instance), 업무(activity), 업무의 수행자(performer), 업무수행시간 등으로 기록한다. 이러한 이벤트로그에서 의미있는 정보와 지식을 추출해내는 프로세스 마이닝은 이미 축적되어있는 트랜잭션 로그 데이터의 분석을 통해 프로세스를 발견하여 업무 프로세스를 개선할 수 있도록 하는 것이다.

우선 <표 1>과 <표 2>의 이벤트로그를 대상으로 선택실행과 동시실행 프로세스를 설명하면 다음과 같다.

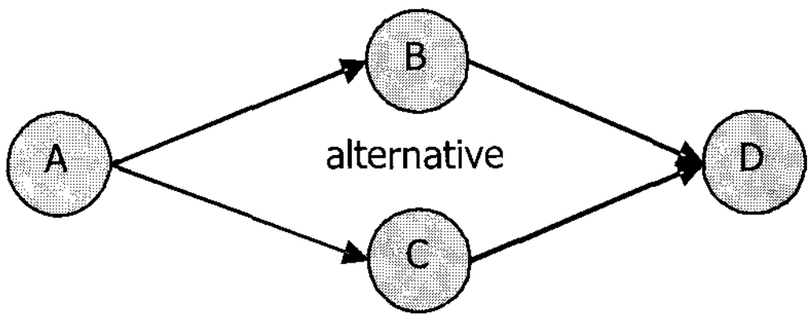
<표 1> 이벤트 로그 I

인스턴스	업무	수행자	업무시간
Case 1	A	홍길동	2006. 02. 01 12 : 00
Case 2	A	홍길동	2006. 02. 01 14 : 00
Case 1	B	김을동	2006. 02. 01 13 : 50
Case 2	C	강나리	2006. 02. 01 16 : 10
Case 1	D	정지훈	2006. 02. 01 16 : 00
Case 2	D	정지훈	2006. 02. 02 09 : 00

<표 2> 이벤트 로그 II

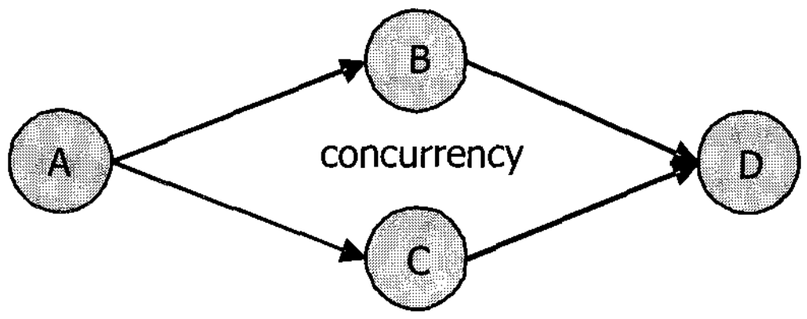
인스턴스	업무	수행자	업무시간
Case 1	A	홍길동	2006. 02. 01 12 : 00
Case 2	A	홍길동	2006. 02. 01 14 : 00
Case 1	B	김을동	2006. 02. 01 13 : 50
Case 1	C	강나리	2006. 02. 01 14 : 00
Case 2	C	강나리	2006. 02. 01 16 : 10
Case 2	B	김을동	2006. 02. 02 09 : 00
Case 1	D	정지훈	2006. 02. 02 09 : 00
Case 2	D	정지훈	2006. 02. 02 13 : 00

<표 1>의 이벤트 로그 I에서 프로세스 인스턴스 기준으로 Case 1은 A→B→D, Case 2는 A→C→D의 순서로 실행되며, 비즈니스 프로세스 상의 업무는 A, B, C, D 네 가지가 있지만 Case 별 트랜잭션을 보면 업무 A, 업무 B 또는 업무 C, 업무 D로 세 개의 업무만을 실행한다. 즉, 각각의 트랜잭션을 보면 프로세스는 업무 A로 시작하고, 업무 A의 수행이 끝나면 업무 B와 업무 C중 하나를 실행한 후 업무 D를 실행하고 종료되는 것을 알 수 있다. 업무 B와 C는 선행업무 A의 실행 후 선택적으로 수행되므로 선택실행 프로세스(alternative route)라 하며, <그림 3>과 같이 표현한다.



<그림 3> 선택실행 프로세스 흐름도

한편, <표 2>의 이벤트 로그 Ⅱ를 보면 Case 1은 A→B→C→D, Case 2는 A→C→B→D 순서로 프로세스가 실행된다. 비즈니스 프로세스 상의 업무는 A, B, C, D 네 가지이고 이 네 가지 업무를 모두 실행하지만, Case별 트랜잭션을 보면 업무의 실행순서가 다른 것을 알 수 있다. 즉, 업무 A로 시작하여 업무 D로 끝나지만, 업무 B와 업무 C는 B→C 혹은 C→B의 순서로 실행되며 업무 B와 C가 프로세스 내에서 병렬적으로 동시에 실행이 된다. 이때 업무 B와 업무 C는 선행업무 A의 실행 후 동시에 병렬적으로 수행되는 동시실행 프로세스(concurrency route)라 하고, <그림 4>와 같이 표현한다.

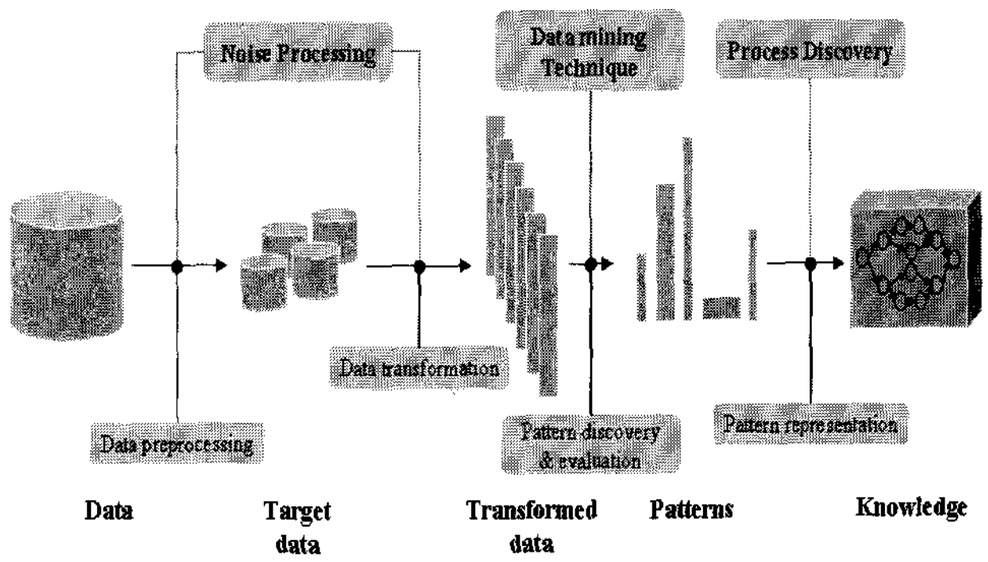


<그림 4> 동시실행 프로세스 흐름도

프로세스 마이닝은 조직모델분석, 기업성과분석, 체계적인 프로세스 수행을 위한 프로세스 분석에 대한 연구가 이루어지는데, 세 가지 관점(업무가 어떻게 진행되는지에 대한 프로세스 관점, 업무 수행자가 누구인지에 대한 조직 관점, 어느 사건에 대한 수행자 또는 프로세스 흐름에 대한 특성이 무엇인지에 대한 사건 관점) 등으로 구분할 수 있으며, 이들 중 프로세스 관점의 연구가 가장 활발히 진행되고 있다.

Agrawal et al.[4]은 워크플로우 관리시스템의 로그를 기반으로 비즈니스 프로세스를 모델링하는 방법을 제시하였으며, Cook and Wolf[5]는 뉴럴 네트워크, 순수 알고리즘, 마코브(Markovian) 방법으로 프로세스 데이터를 분석하여 소프트웨어공학에서 프로세스 모델을 개선하는 연구를 하였고, Herbst[6]는 워크플로우 시스템에 기계학습 알고리즘(Machine learning algorithm)을 이용하여 워크플로우 프로세스 모델을 개선하는 연구를 하였다. 한편, Aalst et al.[3]는 α-algorithm을 이용하여 선택실행 프로세스를 고려한 워크플로우 프로세스 모델을 재발견(rediscovery)하는 연구를 수행하였고, Aalst et al.[1]와 Medeiros et al.[7]은 선택실행 프로세스 뿐 만 아니라 동시실행 프로세스를 고려한 비즈니스 프로세스 개선과 발견을 위해 유전자 알고리즘을 이용하였다. 이들은 유전자 알고리즘을 적용하기 위해 비즈니스 프로세스를 Petri Net으로 표현하여 Casual Matrix를 구성하였고, 페트리넷의 점화규칙을 이용하여 적응도를 평가하였다. 하지만 이미 구조화된 프로세스 모델을 기반으로 Casual Matrix 초기해를 생성하기 때문에 구조화된 프로세스 모델이 없는 경우에는 프로세스 추출이 어렵다. 또한 페트리넷의 점화규칙을 이용하여 도출된 해들의 적응도를 평가하는 시간과 비용(cost)이 매우 높았으며, 데이터양이 증가할수록 알고리즘의 효과성은 급격하게 낮아졌다. 그리고 유전자 알고리즘은 세밀한 해의 탐색이 어렵기 때문에 비즈니스 프로세스 상의 모든 업무와 업무간의 세세한 프로세스 흐름관계를 탐색하는 데는 어려움이 있다.

본 연구에서는 기업의 트랜잭션 로그를 대상으로 동시실행 프로세스, 선택실행 프로세스 그리고 중요한 프로세스이지만 수행 빈도가 낮아 발견되지 못하는 비즈니스 프로세스를 발견하기 위하여 <그림 5>와 같이 데이터 마이닝의 순차패턴과 연관규칙을 이용한 프로세스 마이닝 알고리즘을 개발하고자 한다.



<그림 5> 데이터마이닝을 이용한 프로세스 발견

3. 프로세스 마이닝 알고리즘

기업의 프로세스는 시작업무에서 마지막업무까지 수행되는 순서가 존재하고 여러 서브프로세스(sub-process)들이 포함되어 있으며, 기업의 트랜잭션 로그에는 업무 트랜잭션이 발생할 때 마다 데이터들의 누적이 끊임없이 진행된다. 이렇게 누적되어 있는 기업의 데이터에서 비즈니스 프로세스를 발견하기 위해서는 업무와 업무간의 순서와 연관성을 찾아야 하는데, 대량의 데이터에서 숨겨진 지식, 패턴, 관계를 발견하여 의사결정에 활용할 수 있도록 하는 데이터마이닝 기법 중 연관규칙과 순차패턴은 이를 찾는데 아주 유용한 방법론이다.

연관규칙은 하나의 트랜잭션 내에서 동시에 발생하는 항목간의 연관성을 발견하는 것으로 비즈니스 프로세스 내의 동시에 발생하는 업무를 발견하는 데 유용한 기법이며, 순차패턴은 연관규칙에 시간 변이를 추가한 것으로 트랜잭션 내 항목들의 시간적 관계를 찾아 비즈니스 프로세스 내의 업무 순서를 발견하는데 유용하다. 이러

한 두 기법의 특징을 동시에 고려할 경우 효율적으로 비즈니스 프로세스를 발견할 수 있다.

<표 1>과 <표 2>의 프로세스 인스턴스별 트랜잭션에 연관규칙을 적용하여 프로세스 내의 업무간의 지지도와 신뢰도를 계산하면 <표 3>과 같으며, 연관규칙의 지지도와 신뢰도에 의해 트랜잭션 내의 업무간의 연관성을 파악할 수 있다. 즉, 이벤트 로그 I의 B→C(C→B)의 경우 지지도와 신뢰도 값이 0.0으로서 업무 B와 업무 C 간의 연관성은 없으며, 이를 제외하고는 모두 연관성이 있다고 판단 할 수 있다.

<표 3> 이벤트 로그 Ⅰ, Ⅱ의 업무 간 지지도(S)와 신뢰도(C)

	A→B (B→A)		A→C (C→A)		A→D (D→A)		B→C (C→B)		B→D (D→B)		C→D (D→C)	
	S	C	S	C	S	C	S	C	S	C	S	C
이벤트 로그 I	0.5	0.5	0.5	0.5	1.0	1.0	0.0	0.0	0.5	1.0	0.5	1.0
	0.5	1.0	0.5	1.0	1.0	1.0	0.0	0.0	0.5	0.5	0.5	0.5
이벤트 로그 Ⅱ	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0

주) S(A1→A2) = A1→A2가 포함된 트랜잭션 수/전체 트랜잭션 수
C(A1→A2) = A1→A2가 포함된 트랜잭션 수/A1이 포함된 트랜잭션 수.

<표 1>과 <표 2>에 순차패턴을 적용한 결과는 <표 4>와 같으며, 각 이벤트 로그로부터 프로세스 실행순서를 추출할 수 있음을 알 수 있다.

<표 4> 이벤트 로그 Ⅰ, Ⅱ의 순차패턴

	발견된 순차 패턴
이벤트 로그 I	A→B→D(길이 : 3) A→C→D(길이 : 3)
이벤트 로그 Ⅱ	A→B→C→D(길이 : 4) A→C→B→D(길이 : 4)

연관규칙과 순차패턴의 결과를 동시에 고려해 보면, 이벤트 로그 I의 경우는 순차패턴의 결과로부터 업무 A로 시작하고 업무 A의 수행 후 업무 B 혹은 업무 C가 수행된 후 마지막으로 업무 D가 수행되며, 연관규칙의 결과에서 B→C(C→B)의 지지도와 신뢰도 값이 0.0이므로 업무 B와 업무 C는 모든 트랜잭션에서 서로 독립적인 업무라는 것을 알 수 있다. 즉, 이벤트 로그 I의 경우는 업무 A가 수행된 후 업무 B 또는 업무 C 중 하나를 선택하여 수행하고 마지막으로 업무 D를 수행하는 선택실행 프로세스라는 결론을 도출할 수 있다.

한편, 이벤트 로그 Ⅱ의 경우는 순차패턴의 결과로부터 업무 A로 시작하고 업무 A가 수행된 후 업무 B와 업무 C 혹은 업무 C와 업무 B가 실행되고 마지막으로 업무 D가 수행되며, 연관규칙의 결과에서 B→C(C→B)의 지지도와 신뢰도 값이 1.0이므로 업무 B와 업무 C는 연관성이 있는 업무로 정방향 뿐만 아니라 역방향의 지지도와 신뢰도를 만족하는 것을 알 수 있다. 즉, 이벤트 로그 Ⅱ의 경우는 업무 A 수행 후 업무 B와 업무 C를 실행(B→C 또는 C→B)하고, 이 두 업무(B와 C)가 모두 실행된 후 업무 D를 실행하는 동시실행 프로세스라는 결론을 도출할 수 있다.

따라서 본 연구에서는 순차패턴과 연관규칙을 동시에 고려하여 이벤트로그에서 동시실행 프로세스, 선택실행 프로세스 그리고 중요한 프로세스이지만 수행 빈도가 낮아 발견되지 못하는 비즈니스 프로세스를 발견하는 알고리즘을 개발하였다.

순차패턴과 연관규칙을 동시에 고려한 프로세스 마이닝의 알고리즘을 단계별로 기술하면 다음과 같다.

Step 1 : 이벤트로그에서 인스턴스별 업무의 실행순서를 탐색한다.

Step 2 : 최소지지도를 만족하는 시작업무를 추출하고 프로세스의 start activity로 설정한다.

Step 3 : Step 1에서 탐색한 순서에서 이웃한 업무 X와 Y의 정방향(X→Y)과 역방향(Y→X) 지지도와 신뢰도를 계산한다.

Step 4 : 프로세스에 연결된 업무(activity X)와 인접하는 업무이지만 아직 연결되지 않은 업무(activity Y)들의 정방향(X→Y) 지지도와 신뢰도가 최소 지지도 및 신뢰도를 만족하면 연결한다.

Step 5 : 만약 업무 X와 연결된 업무가 2개 이상이라면 선택실행 프로세스로 설정한다.

Step 6 : 프로세스에 연결된 업무(activity X)와 인접하는 업무(activity Y)가 이미 프로세스에 연결되어 있을 경우, 역방향(Y→X)의 지지도와 신뢰도가 최소 지지도 및 신뢰도를 만족하면 업무 X와 Y를 서로 동시에 실행되는 동시실행 프로세스로 설정하고, 그렇지 않다면 단순히 연결만 한다.

Step 7 : 모든 업무가 연결될 때까지 Step 4에서 Step 6까지를 반복 실행한다.

<그림 6>은 본 연구의 알고리즘을 자바 프로그램 언어의 문법에 따라 표현한 것이다.

처음 initialize() 메소드에서는 업무 트랜잭션 로그파일에서 각 프로세스 인스턴스의 트랜잭션을 생성한다. 다음 sequenceDiscoverd() 메소드에서는 인스턴스별 업무의 실행순서를 탐색한다.

두 번째 단계의 startExtract() 메소드에서는 트랜잭션의 맨 처음 업무의 지지도를 계산하여 최소지지도를 만족하는 업무를 도출하여 시작업무로 한다. rightCalculator()와 leftCalculator() 메소드에서는 sequenceDiscoverd() 메소

드에서 추출된 인스턴스별 업무 실행순서에서 길이가 2인 빈번한 패턴을 추출하여 정방향과 역방향 지지도와 신뢰도를 계산한다.

ProcessFlow() 메소드에서는 최소 지지도와 최소 신뢰도를 만족하는 업무들을 연결하여 기본적인 프로세스를 도출하고, 역방향의 최소 신뢰도를 만족하는지의 여부에 따라 선택실행과 동시실행을 설정한다. 여기서 최소 지지도와 최소신뢰도 값은 0.0~0.3사이의 값을 임의로 설정한다. 만약 최소지지도와 최소신뢰도를 너무 작게 하거나 크게 하면 잘못된 트랜잭션을 도출할 수도 있다. 마지막으로 추출된 결과에 따라 프로세스의 흐름을 도출하고 종료한다.

```
public void BPMining( ) {
        int num ;
        initialize( ) ;
        for (num = 0 ; num<eventlogSize ; num++) {
                sequenceDiscoverd( ) ;
        }
        for (num = 0 ; num < sequenceSize ; num++) {
                startExtract( ) ;
                rightCalculator( ) ;
                leftCalculator( ) ;
        }
        ProcessFlow( ) ;
}
 public void ProceeFlow() {
        double minSup = MinSupport( ) ;
        double minCon = MinConfidence( ) ;
        for (num = 0 ; num < sequenceSize ; num++) {
                        FlowConnection( ) ;
        }
}
```

<그림 6> 본 연구의 알고리즘

4. 실험 및 비교분석

4.1 알고리즘의 실험결과

본 연구에서 제시한 알고리즘의 실험을 수행하기 위하여 <표 5>와 같이 업무 수 7개, 프로세스 인스턴스 수 9개, 로그 수는 38개로 로그 파일을 생성하였다. 그리고, 최소 지지도와 신뢰도의 값은 0.15로 설정하였다.

알고리즘의 첫 번째 단계로 로그파일에서 인스턴스별 업무의 실행순서를 탐색하면 A→B→C→D→F→G, A→E→G, A→E→G, A→B→D→C→F→G, B→G, A→B→C→D→F→G, A→E→G, A→E→G, A→B→D→C→F→G의 업무 실행 순서가 도출된다.

두 번째 단계에서 프로세스 시작업무를 추출하게 되는데, 시작업무의 대상은 인스턴스별 업무의 실행순서 중 맨 처음 실행되는 업무들이 되며 이들 업무 중 최소 지지도를 만족하는 업무가 시작업무가 된다. 즉, 시작업무의 대상 업무는 업무 A와 업무 B가 되고, 이중 최소 지지도(0.15)를 만족하는 업무 A(지지도 0.889)가 시작업무가 된다.

세 번째 단계에서는 Step 1에서 탐색한 업무실행 순서에서 이웃한 업무들을 도출해야 되는데, 이는 길이가 2인 순차패턴을 실행하면 된다. 실행결과 A→B, A→E, B→C, B→D, B→G, C→D, C→F, D→C, D→F, E→G, F→G가 추출되고, 추출된 업무 X와 Y의 정방향(X→Y)과 역방향(Y→X) 지지도와 신뢰도를 계산하면 <표 6>과 같은 결과를 얻을 수 있다.

<표 5> 이벤트 로그 Ⅲ

인스턴스	업무	수행자	업무시간
Case 1	A	홍길동	2006. 02. 01 09 : 00
Case 6	A	배상환	2006. 02. 01 09 : 05
Case 2	A	김수자	2006. 02. 01 09 : 15
Case 7	A	김필승	2006. 02. 01 09 : 30
Case 3	A	홍길동	2006. 02. 01 11 : 45
Case 8	A	배상환	2006. 02. 01 11 : 50
Case 1	B	정지훈	2006. 02. 01 12 : 00
Case 6	B	김 별	2006. 02. 01 13 : 00
Case 4	A	김수자	2006. 02. 01 13 : 30
Case 9	A	김필승	2006. 02. 01 13 : 30
Case 2	E	이주근	2006. 02. 01 15 : 00
Case 7	E	이상성	2006. 02. 01 15 : 00
Case 5	B	정지훈	2006. 02. 01 15 : 45
Case 1	C	홍정만	2006. 02. 01 16 : 00
Case 6	C	이빛나	2006. 02. 01 16 : 00
Case 4	B	정지훈	2006. 02. 01 17 : 00
Case 5	G	강나리	2006. 02. 01 17 : 00
Case 9	B	김 별	2006. 02. 01 17 : 00
Case 2	G	강나리	2006. 02. 02 09 : 00
Case 7	G	문우섭	2006. 02. 02 09 : 00
Case 3	E	이주근	2006. 02. 02 09 : 30
Case 8	E	이상성	2006. 02. 02 09 : 30
Case 1	D	김을동	2006. 02. 02 10 : 00
Case 6	D	송 빈	2006. 02. 02 10 : 00
Case 3	G	강나리	2006. 02. 02 13 : 00
Case 8	G	문우섭	2006. 02. 02 13 : 00
Case 1	F	김수자	2006. 02. 02 14 : 00
Case 6	F	김필승	2006. 02. 02 14 : 00
Case 4	D	김을동	2006. 02. 02 14 : 30
Case 9	D	송 빈	2006. 02. 02 14 : 30
Case 4	C	홍정만	2006. 02. 03 09 : 00
Case 9	C	이빛나	2006. 02. 03 09 : 00
Case 4	F	김수자	2006. 02. 03 12 : 00
Case 9	F	김필승	2006. 02. 03 12 : 00
Case 1	G	강나리	2006. 02. 03 13 : 00
Case 6	G	문우섭	2006. 02. 03 13 : 00
Case 4	G	강나리	2006. 02. 03 16 : 45
Case 9	G	문우섭	2006. 02. 03 16 : 45

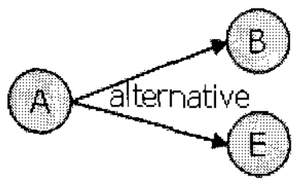
<표 6> 이웃한 업무간의 지지도와 신뢰도

	순차 업무	정방향		역방향	
		지지도	신뢰도	지지도	신뢰도
1	A→B	0.4	0.5	0.0	0.0
2	A→E	0.4	0.5	0.0	0.0
3	B→C	0.2	0.4	0.0	0.0
4	B→D	0.2	0.4	0.0	0.0
5	B→G	0.1	0.2	0.0	0.0
6	C→D	0.2	0.5	0.2	0.5
7	C→F	0.2	0.5	0.0	0.0
8	D→F	0.2	0.5	0.0	0.0
9	E→G	0.4	1.0	0.0	0.0
10	F→G	0.4	1.0	0.0	0.0

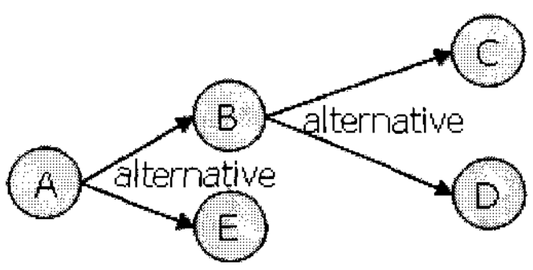
Step 4에서 Step 7까지의 반복실행 중 단계별 프로세스 도출 결과는 다음과 같다.

- *iteration 0* : 두 번째 단계에서 도출된 업무 A로 시작

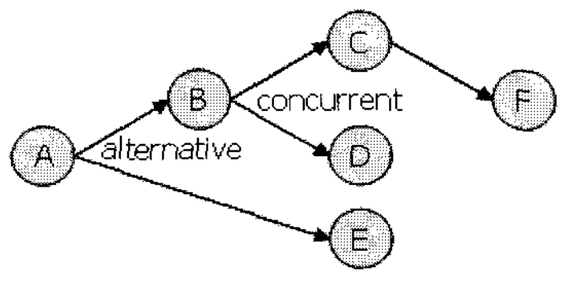
- *iteration 1* : 업무 A와 인접하지만 연결되지 않은 업무 B와 E가 정방향 최소 지지도와 신뢰도를 만족하므로 업무 A와 업무 B, E 를 연결하고 선택실행으로 설정



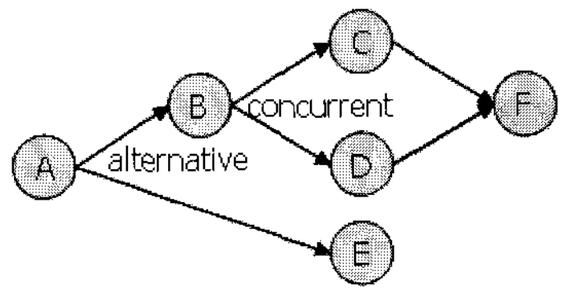
- *iteration 2* : 업무 B와 인접하지만 연결되지 않은 업무 C, D, G 중 C와 D가 정방향 최소 지지도와 신뢰도를 만족하므로 업무 B와 업무 C, D를 연결하고 선택실행으로 설정



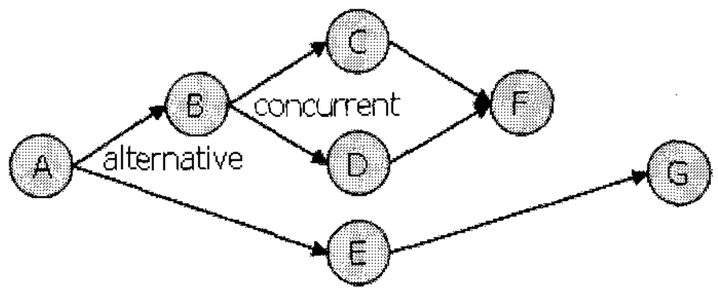
- *iteration 3* : 업무 C와 인접하지만 연결되지 않은 업무 F가 정방향의 최소지지도와 신뢰도를 만족하므로 우선 연결하고, 정방향의 최소지지도와 신뢰도를 만족하면서 인접한 업무 D는 이미 프로세스에 연결되어 있고 역방향 최소 지지도와 신뢰도를 만족하므로 동시실행으로 설정



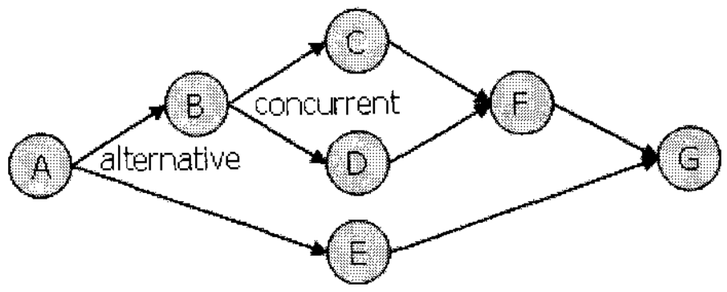
- *iteration 4* : 업무 D와 인접하면서 정방향의 최소지지도와 신뢰도를 만족하는 업무 F는 이미 프로세스에 연결되어 있고, 역방향의 최소지지도와 신뢰도를 만족하지 않으므로 단순히 연결만 함



- *iteration 5* : 업무 E와 인접하면서 정방향의 최소지지도와 신뢰도를 만족하는 업무 G를 연결



- *iteration 6* : 업무 F와 인접하면서 정방향의 최소지지도와 신뢰도를 만족하는 업무 G는 이미 프로세스에 연결되어 있고, 역방향의 최소지지도와 신뢰도를 만족하지 않으므로 단순히 연결만 함



모든 업무가 연결되었으므로 알고리즘은 종료되고, iteration 6에서 얻어진 프로세스가 <표 6>의 이벤트 로그 Ⅲ로부터 연관규칙과 순차패턴을 적용하여 발견된 프로세스의 결과이다.

4.2 성능평가를 위한 비교실험

본 연구에서 제시한 알고리즘의 효과성과 효율성을 평가하기 위하여 Medeiros et al.[7]이 제시한 유전자 알고리즘과 본 연구의 알고리즘을 비교 분석하였다. 이를 위하여 Aalst et al.[1]이 제시한 이벤트 로그 <표 7>을 대상으로 자바 프로그래밍 언어로 알고리즘들을 구현하여 Pentium4-2.93GHz의 윈도우즈 2000 서버 운영체제에서 실험하였다.

<표 7>을 대상으로 Medeiros et al.[7]이 제시한 유전자 알고리즘과 본 연구의 알고리즘으로 도출한 프로세스의 결과는 <그림 7>과 같이 동일하게 나타났다. 그러나, 동일한 이벤트 로그로 각 알고리즘을 반복 실험한

결과 수행시간과 프로세스 일치도의 평균값에 있어서는 차이가 나타났으며, 실행 결과는 <표 8>과 같고, 본 연구의 알고리즘이 Medeiros et al.[7]의 알고리즘에 비해 수행시간과 프로세스 일치도에서 좋은 결과를 보여주고 있음을 알 수 있다. 여기서,

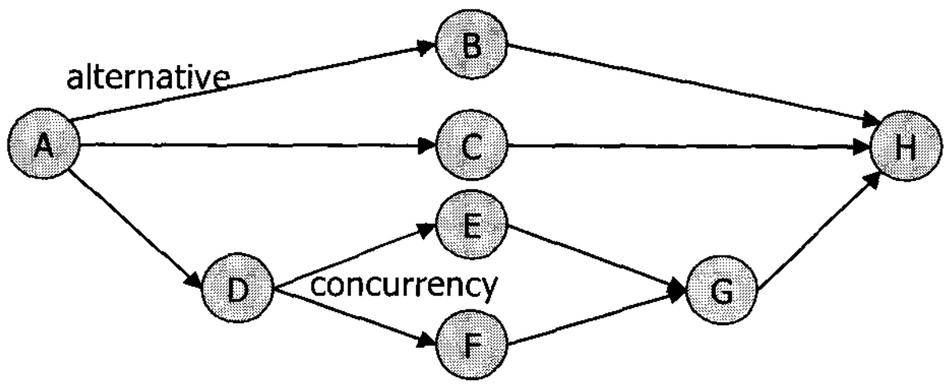
$$\text{프로세스 일치도} = 1 - \frac{MissingRelationOfActivities}{\sum_{i=1}^{n}(n-1)}$$

n = 업무의 수

$MissingRelationOfActivities$ = 업무간 잘못 연결된 개수

〈표 7〉 Aalst et al.[1]의 이벤트 로그 예

Case	업무	수행자	업무시간
Case 1	A	John	9-3-2004 : 15.01
Case 2	A	John	9-3-2004 : 15.12
Case 3	A	Sue	9-3-2004 : 16.03
Case 3	D	Carol	9-3-2004 : 16.07
Case 1	B	Mike	9-3-2004 : 18.25
Case 1	H	John	10-3-2004 : 9.23
Case 2	C	Mike	10-3-2004 : 10.34
Case 4	A	Sue	10-3-2004 : 10.35
Case 2	H	John	10-3-2004 : 12.34
Case 3	E	Pete	10-3-2004 : 12.50
Case 3	F	Carol	11-3-2004 : 10.12
Case 4	D	Pete	11-3-2004 : 10.14
Case 3	G	Sue	11-3-2004 : 10.44
Case 3	H	Pete	11-3-2004 : 11.03
Case 4	F	Sue	11-3-2004 : 11.18
Case 4	E	Clare	11-3-2004 : 12.22
Case 4	G	Mike	11-3-2004 : 14.34
Case 4	H	Clare	11-3-2004 : 14.38



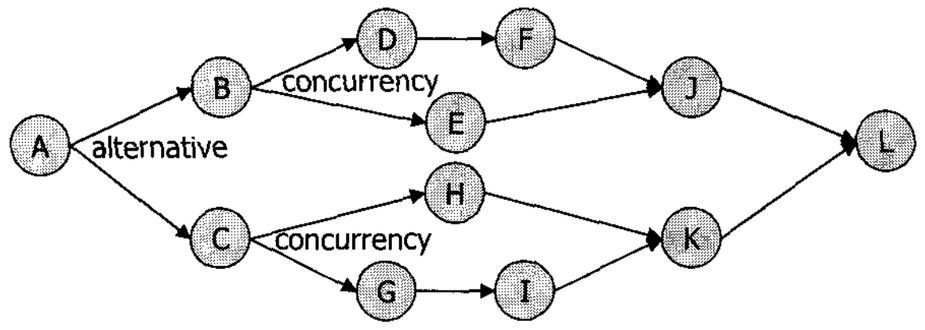
〈그림 7〉 〈표 7〉의 프로세스 결과

그 이유로 Medeiros et al.[7]의 알고리즘은 casual matrix의 초기해가 어떻게 생성되는지에 따라 해의 정확성과 탐색 시간이 달라지고, 해의 적응도를 평가하기 위한 시간도 많이 걸려 세밀한 해의 탐색이 이루어 지지 않아 동시실행 프로세스나 선택실행 프로세스를 완벽하게 추출하지 못하기 때문이다. 반면 본 연구는 업무와 업무 간의 순서와 연관성을 동시에 고려하여 프 로세스에서 업무의 실행 순서와 프로세스 실행에 있어서의 업무들의 관계에 따라 프로세스를 탐색하였기 때문에 동시실행과 선택실행 프로세스를 보다 효과적이고 효율적으로 추출할 수 있었다.

〈표 8〉 반복실험 결과비교

event log수 /case 수	Medeiros et al.[7]		본 연구	
	프로세스 일치도	시간(초)	프로세스 일치도	시간(초)
18/4	99.3%	7.02	100%	0.44

위의 결과에 대한 타당성을 검증하기 위하여 <그림 8>과 같은 보다 복잡한 프로세스를 갖는 로그를 생성한 후, 각 알고리즘을 실행시킨 결과는 <표 9>와 같다.



〈그림 8〉 타당성 검증을 위한 프로세스

〈표 9〉 반복실험 결과비교

event log수 /case 수	Medeiros et al.[7]		본 연구	
	프로세스 일치도	시간(초)	프로세스 일치도	시간(초)
29/4	88.7%	475.67	100%	0.47

<표 9>의 결과로부터 데이터양이 많아지고 프로세스가 복잡해질수록 Medeiros et al.[7]의 연구는 초기해와 해의 적응도 평가의 문제점으로 인해 세밀한 해의 탐색이 이루어 지지 않아 동시실행 프로세스나 선택실행 프로세스를 완벽하게 추출하지 못하였을 뿐만 아니라, 유전자의 적응도를 평가할 때마다 트랜잭션 로그를 계속해서 탐색하기 때문에 알고리즘 수행시간이 급격하게 증가함을 알 수 있었다. 반면에 본 연구에서 제시한 알고리즘은 첫 번째 단계에서 트랜잭션 로그파일에 있는 데이터를 탐색하여 인스턴스별 업무 실행순서를 추출한 후 순차패턴과 연관규칙을 이용한 알고리즘을 수행하기 때문에 알고리즘의 수행시간이 로그파일의 크기에 크게

영향 받지 않아 Medeiros et al.[7]의 알고리즘보다 더 효율적이고도 효과적인 결과를 얻을 수 있었다.

5. 결 론

본 연구는 기업의 트랜잭션 로그를 대상으로 동시실행 프로세스, 선택실행 프로세스 그리고 중요한 프로세스이지만 수행 빈도가 낮아 발견되지 못하는 비즈니스 프로세스를 발견하기 위해 데이터 마이닝의 순차패턴과 연관규칙을 이용한 프로세스 마이닝 알고리즘을 제시하였다.

본 논문에서 제시한 알고리즘의 성과측정을 위하여 Medeiros et al.[7]의 알고리즘과 비교분석을 실시하였고, 데이터양이 많아지고 프로세스가 복잡해질수록 정확도와 수행시간에 있어 Medeiros et al.[7]에 비해 본 연구의 알고리즘이 동시실행과 선택실행 프로세스를 보다 효과적이고 효율적으로 추출할 수 있음을 보여주었다.

향후 연구에서는 루프백(loop-back)을 고려한 비즈니스 프로세스를 발견하는 연구를 수행할 계획이며, 이와 함께 비즈니스 프로세스의 유기적인 리소스를 통합하는 연구를 계속하고자 한다.

참고문헌

[1] Aalst, W. M. P. van der, Medeiros, A. K. A. de, and Weijters A. J. M. M.; "Genetic Process Mining," *Lecture notes in computer science*, 3536 48-69, 2005.

[2] Aalst, W. M. P. van der, Reijers, H. A., Weijters, A. J. M. M., Dongen, B. F. van, Alves de Medeiros, A. K., Song, M. S., and Verbeek, H. M. W.; "Business process mining : An industrial application," *Information systems*, 32(5) : 713-732, 2007.

[3] Aalst, W. M. P. van der, Weijters, A. J. M. M., and Maruster, L.; "Workflow Mining : Discovering Process Models from Event Logs," *IEEE Transactions on Knowledge and Data Engineering*, 16(9) : 1128-1142, 2004.

[4] Agrawal, R., Gunopulos, D., and Leymann, F.; "Mining Process Models from Work-flow Logs," *In 6th International Conference on Extending Database Technology*, 469-483, 1998.

[5] Cook, J. E. and Wolf, A. L.; "Discovering Models of Software Processes from Event Based Data," *ACM Transactions on Software Engineering and Methodology*, 7 (3) : 215-249, 1998.

[6] Herbst J.; "A Machine Learning Approach to Workflow Management," *Lecture Notes in Computer Science*, 1810 : 183-194, 2000.

[7] Medeiros, A. K. A. de, Weijters A. J. M. M., and Aalst, W. M. P. van der; "Genetic Process Mining : A Basic Approach and Its Challenges," *Lecture Notes In Computer Science*, 3812 : 203-215, 2006.