# PROCL : 프로세스 로그 클러스터링 시스템

## PROCL : A Process Log Clustering System

정 재 윤(Jae-Yoon Jung)*


### 초 록

프로세스 마이닝은 프로세스 실행 결과로부터 유용한 프로세스 정보를 추출하는 연구이다. BPMS, ERP, SCM 등 프로세스 인식 정보시스템들이 확산되면서 프로세스 마이닝 연구가 더욱 활발해지고 있다. 본 논문에서는 프로세스 마이닝 이전에 먼저 프로세스 로그를 군집화하는 방법과 구현 시스템을 제시한다. 본 연구의 프로세스 로그 클러스터링은 기존에 제시된 여러 가지 프로세스 마이닝 알고리즘들과 함께 사용함으로써 프로세스 마이닝의 과정을 개선시킬 수 있다. 프로세스 클러스터링 시스템은 분석 요구에 따라 적절한 개수의 프로세스 로그로 군집화함으로써 사용자가 원하는 수준의 프로세스 모델들을 추출하도록 지원한다. 프로세스 마이닝 오픈 툴인 ProM 플랫폼을 바탕으로 하여 본 논문에 제시된 프로세스 클러스터링 기법을 적용하고 개발하였다.

### ABSTRACT

Process mining aims at extracting useful information from system log of business process execution. As process aware information systems, such as BPMS, ERP, and SCM, spread, researches on process mining get more significance. In this paper, we propose the methodology of clustering process log before process mining and also present the prototype system. The proposed methodology can be used in accompany with the existing process mining algorithms to improve their performance. The process log clustering system PROCLE, presented in this paper, supports to classify the process instances in the system log in order to extract the appropriate level of process model according to the users' need. The proposed methodology was implemented on the open platform for process mining, ProM.


**키워드** : 프로세스 마이닝, 프로세스 유사성, 로그 클러스터링
Process Mining, Process Similarity, Log Clustering

---

이 논문은 한국학술진흥재단의 지원을 받아 수행되었음(KRF 2005 214 D00397).

* 경희대학교 산업공학과 전임강사, 산학협력기술연구원

# 1. 서 론

비즈니스 프로세스 관리 기술은 프로세스 자동화 및 신행 관리로부터 출발하여[6, 8], EIP(Enterprise Information Portal), BAM (Business Activity Monitoring), RTE(Real-Time Enterprise) 등 실시간 정보공유 및 모니터링을 지원하는 시스템으로 발전하였으며 [4], 최근에는 프로세스 마이닝(process mining)[15] 등 프로세스의 분석과 개선을 위한 연구로 발전하고 있다.

프로세스 마이닝은 프로세스 실행 결과로부터 유용한 프로세스 정보를 추출하는 연구이며, BPM, ERP, SCM 등의 기업 정보화를 위하여 프로세스 인식 가능한 정보시스템들이 확산되면서 프로세스 마이닝 연구가 더욱 활발해지고 있다[5, 17]. 본 논문에서는 프로세스 마이닝 과정을 개선하기 위하여 프로세스 로그를 분석하기 이전에 먼저 군집화하는 프로세스 로그 클러스터링에 관한 접근법을 제안하고, 그 구현 시스템을 제시한다.

프로세스 로그 클러스터링 시스템인 PROCL은 분석 요구에 따라 적절한 수의 프로세스 로그로 군집화함으로써 사용자가 원하는 수준의 프로세스 모델들을 추출할 수 있는 방안을 제공한다. 특히, 프로세스 로그 내의 프로세스 인스턴스들은 액티비티 벡터와 트렌지션 벡터로 변환된 후, 유사성 측도에 의해 비교되어 군집화된다. 본 연구는 기존에 제시된 여러 가지 프로세스 마이닝 알고리즘들[3, 5, 10, 14]과 연계하여 사용할 수 있도록 지원함으로써, 프로세스 마이닝의 성능을 향상시킬 수 있다.

본 논문의 구성은 다음과 같다. 제 2장에서는 먼저 프로세스 분석 및 개선에 관련된 연구들을 살펴본다. 제 3장에서는 프로세스 로그 클러스터링의 접근법을 소개하고, 제 4장에서는 이 접근법을 반영하여 구현된 시스템을 제시한 후, 기존의 프로세스 마이닝 알고리즘과 연계하여 사용하는 과정을 보여준다. 마지막으로 제 5장에서는 본 논문에 대한 토의와 함께 결론을 맺는다.

# 2. 관련 연구

프로세스 마이닝은 기업정보시스템에서 발생하는 프로세스 실행 로그를 바탕으로 의미있는 정보를 추출해내는 과정이다. 프로세스 저장소의 실행 결과 및 이벤트에 관한 로그를 추출하여 기존의 통계적 기법, 인공지능 알고리즘, 소셜 네트워크 기법 등을 통하여 분석함으로써[3, 5, 10, 14], 업무의 의존 관계를 표현할 수 있는 프로세스 모델을 유도하거나, 업무의 상관관계, 작업자의 업무 전달 관계 등의 프로세스 수행 상의 특징을 분석한다 [15]. 프로세스 마이닝에 관한 여러 가지 연구들은 아인트호벤 공대에서 개발한 오픈 프레임워크인 ProM 시스템을 통해 플러그인 (plug in) 형태로 구현되었는데[16], 이 플랫폼은 ERP나 BPM 시스템의 프로세스 로그를 임포트하여 프로세스 모델 추출, 프로세스 모델 분류, 프로세스 패턴 추출, 작업자간 업무 전달 형태 등을 분석할 수 있는 환경을 제공한다. 본 연구에서 제시된 프로세스 로그 클러스터링 기법도 ProM 플랫폼 상에 적용함으로써 입력 데이터 포맷을 공유하고, 다른 알고리즘과 연계할 수 있도록 구현되었다.

본 연구의 프로세스 로그 클러스터링을 위하여 사용된 방법은 프로세스 로그 구조를 분석하여 프로세스 유사성 척도를 사용하여 군집화하는 것이다. 기존의 비즈니스 프로세스 설계 및 개선 연구에서 프로세스의 구조적 분석을 수행한 연구로는 프로세스 상속, 프로세스 비교 척도, 프로세스 평가 척도 등이 있다. 프로세스 상속에 관한 연구[7, 13]는 프로세스 간의 종속 관계를 분석함으로써, 기업간 프로세스 모델로 확장하거나 변화 관리를 지원하기 위하여 제시되었다. 한편, 프로세스 비교 척도에 관한 연구[1]는 신규 프로세스를 공동 설계할 때, 다수의 전문가가 설계한 프로세스 구조를 비교하여 하나의 통합 프로세스 모델을 도출하기 위하여 수행되었다. 한편, 최근에는 프로세스 모델의 안정성과 견고함을 평가하기 위한 프로세스 구조 분석에 관한 연구들이 진행되고 있다. 프로세스 복잡도 척도는 프로세스 모델의 분기와 병합에 대한 복잡도를 평가하여 선행 오류를 줄이기 위한 척도로서 제안되었으며[2], 프로세스 모델의 응집성 및 결합성 척도는 제품 정보와 같이 프로세스 모델에 포함된 정보의 전달 방식을 평가함으로써 응집성과 결합성이라는 두 척도에 대한 균형적인 프로세스 모델을 도출하기 위하여 제안되었다[9]. 이처럼 프로세스 상속, 비교 척도, 평가 척도를 비롯한 프로세스 구조 분석에 관한 연구는 프로세스의 구조적 분석을 통하여 동적 변화 가능성을 탐색하고, 통합 모델을 도출하거나, 적정 모델을 선택하는 데 도움을 줌으로써 프로세스 설계를 지원하기 위하여 다양하게 시도되었다. 본 논문에서 제안하는 프로세스 유사성에 의한 클러스터링은 프로세스의 구조를 분석한다는 점에서 프로세스 상속 및 평가 척도와 같은 성격을 지니고 있지만, 프로세스 구조를 바탕으로 프로세스 로그를 클러스터링 하기 위하여 제시되었다.

## 3. 프로세스 로그 클러스터링

프로세스 로그 클러스터링은 시간에 따라 순차적으로 기록된 수많은 프로세스 인스턴스들의 선행 기록을 분석 대상으로 한다. 프로세스 인스턴스란 하나의 프로세스를 실제로 선행한 결과로서 프로세스 케이스(case)라고도 부르며, 하나의 프로세스 인스턴스는 그 프로세스를 실행하는 데 참여한 액티비티들의 실행 이벤트를 포함하고 있다. 프로세스 로그는 다양한 프로세스 모델에서 수행된 프로세스 인스턴스들이 혼재되어 있다고 가정한다.

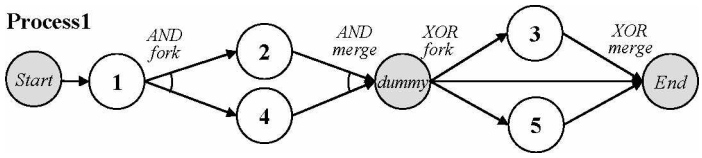
본 연구에서는 프로세스 로그 클러스터링을 위하여 하나의 프로세스 인스턴스를 액티비티 벡터와 트랜지션 벡터로 표현한다. 두 가지 벡터에 기반으로 프로세스 인스턴스들은 로그 클러스터를 군집화하는데, 계층적 클러스터링 방법(Agglomerative Hierarchical Clustering)을 사용한다. 이 수행 과정들은 크게 (1) 프로세스 로그 벡터 생성, (2) 유사성 측정, (3) 클러스터링의 3단계로 구분할 수 있다.

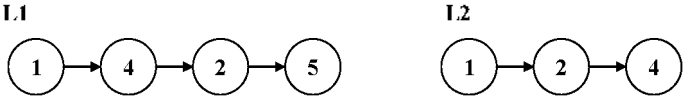
### 3.1 프로세스 로그 벡터(process log vector) 생성

프로세스 로그에 기록된 누적 정보인 프로세스 인스턴스 간의 유사성을 비교하는 데

사용하기 위하여, 본 연구에서는 각 프로세스 인스턴스 $L$을 두 가지 형태의 프로세스 로그 벡터로 표현한다. 먼저, 액티비티 벡터(activity vector) $A(L)$는 프로세스 수행 과정에서 액티비티의 실행 유무를 표현하는 수단으로, 두 프로세스가 공통적으로 실행된 액티비티 수를 측정하는 데 사용된다. 즉, 중복되는 액티비티가 많을수록 더 유사한 프로세스 인스턴스라고 간주한다. 다음으로, 트랜지션 벡터(transition vector) $T(L)$는 하나의 프로세스 인스턴스 내에서 실행된 액티비티들의 선후 관계가 얼마나 유사한지 비교하는 수단으로서 두 가지 유형의 벡터로 다시 구분된다. 명시적 트랜지션(explicit transition) 벡터 $T^{ex}(L)$는 인접한 두 액티비티의 직접적인 선후관계만을 저장하며, 유추적 트랜지션(implicit transition) 벡터 $T^{im}(L)$는 인접 액티비티 간의 선후관계뿐만 아니라, 비인접 액티비티간의 선후관계도 그 거리에 반비례한 가중치를 적용하여 유사성을 측정한다. 프로세스 로그 내의 프로세스 인스턴스를 표현하는 세 가지 벡터는 아래와 같이 표현할 수 있다. $i$, $j$는 액티비티를 나타내는 계수이며, $n$은 프로세스 로그에 포함될 수 있는 액티비티의 총 개수이다.

$$A(L) = (A_i), \quad A_i = 0 \text{ or } 1 \quad where\ i = 1, \cdots, n$$
$$T^{ex}(L) = (T_{ij}), \quad T_{ij} = 0 \text{ or } 1 \quad where\ i, j = 1, \cdots, n$$
$$T^{im}(L) = (T_{ij}), \quad T_{ij} = 1/d_{ij} \quad where\ i, j = 1, \cdots, n$$




| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 1/2 | 0 | 1 | 1/3 |
| 2 | 0 | 0 | 0 | 0 | 1 |
| 3 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 1 | 0 | 0 | 1/2 |
| 5 | 0 | 0 | 0 | 0 | 0 |

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1/2 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 |

$A(L1) = (1, 1, 0, 1, 1)$
$T^{ex}(L1) = (0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0)$
$T^{im}(L1) = (0, 1/2, 0, 1, 1/3, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1/2, 0, 0, 0, 0)$

$A(L2) = (1, 1, 0, 1, 0)$
$T^{ex}(L2) = (0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
$T^{im}(L2) = (0, 1, 0, 1/2, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$

〈Figure 1〉 Example of process log vectors

<Figure 1>은 예제 프로세스 **Process 1**을 실행한 결과에서 나타날 수 있는 프로세스 로그 중 두 가지 경우(1→4→2→5, 1→2→4)의 액티비티 벡터 $A(L)$, 명시적 트랜지션 벡터 $T^{ex}(L)$, 유추적 트랜지션 벡터 $T^{im}(L)$를 보여준다.

### 3.2 유사성 측정(similarity measurement)

두 개의 프로세스 로그 벡터를 비교하여 유사성을 측정하는 측도(metrics)로 본 연구에서는 Jaccard 계수와 Cosine 계수를 사용한다. 각 계수는 두 프로세스의 액티비티 벡터, 또는 트랜지션 벡터를 사용하여 프로세스 인스턴스 간의 유사성을 정량적으로 비교하는 측도를 제공한다.

Jaccard 계수는 두 벡터에서 공통으로 발생하는 요소, 즉, 액티비티나 트랜지션의 수를 두 벡터의 합집합의 크기로 나눈 값이다[12]. 이 계수는 이진 벡터(binary vector)의 비교를 위해서만 사용될 수 있어서, 유추적 트랜지션과 같이 실수 벡터(numeric vector)에는 사용할 수 없다. 아래는 액티비티 벡터의 Jaccard 계수 계산식을 보여주며, 트랜지션 벡터도 동일하게 계산된다.

$$Sim_{Jaccard}(A(L1), A(L2)) = \frac{|A_{L1} \cap A_{L2}|}{|A_{L1} \cup A_{L2}|}$$

<Figure 1>의 예에서 액티비티 유사성은 $sim_{Jaccard}(A(L1), (A(L2)) = 3/4$이며, 명시적 트랜지션 유사성은 $sim_{Jaccard}(T^{ex}(L1), T^{ex}(L2)) = 0/5 = 0$로 계산된다. 공통된 액티비티는 3개이지만, 공통된 인접 트랜지션은 존재하지 않기 때문이다.

Cosine 계수는 이진 벡터나 실수 벡터의 모든 경우에 사용될 수 있는데, 벡터 공간에서 두 벡터의 사이각 크기의 Cosine 값과 동일하다[11]. 특히, Cosine 계수는 비중심 상관계수(uncentered correlation coefficient)와 동일한 역할을 한다. 트랜지션 벡터는 아래와 같이 계산되며, 액티비티 벡터도 마찬가지로 계산된다.

$$sim_{Cosine}(T(L1), T(L2)) = \frac{T_{L1} \cdot T_{L2}}{|T_{L1}||T_{L2}|}$$

<Figure 1>의 예에서 액티비티 유사성은 $sim_{Cosine}(A(L1), A(L2)) = 3/\sqrt{4}\sqrt{3} = \sqrt{3}/2$이며, 명시적 트랜지션 유사성은 $sim_{Cosine}(T^{ex}(L1), T^{ex}(L2)) = 0/\sqrt{3}\sqrt{2} = 0$, 유추적 트랜지션 유사성은 $sim_{Cosine}(T^{im}(L1), T^{im}(L2)) = (1/2 + 1/2)/\sqrt{1*3+(1/4)*2+1/9}\sqrt{1*2+1/2} = 0.287$로 계산된다. 공통된 액티비티는 세 개이며, 공통된 인접 트랜지션은 없지만, 1/2의 값을 가지는 유추적 트랜지션이 공통으로 2개가 존재하기 때문이다.

### 3.3 로그 클러스터링 알고리즘 (Clustering algorithms)

여러 프로세스 로그들의 벡터와 유사성 척도를 사용하여 모든 프로세스들 간의 유사성을 측정한 후에 로그 클러스터링을 수행한다. 본 연구에서 개발된 시스템에서는 두 가지 클러스터링 알고리즘을 사용하였는데, Agglomerative Hierarchical Clustering(AHC) 알

고리즘과 K means 알고리즘을 수행할 수 있도록 지원하였다. 세 가지 프로세스 로그 벡터들은 각 유사성에 따라 별도로 클러스터링을 수행할 수도 있으며, 액티비티 벡터와 트랜지션 벡터의 유사성들을 가중평균하여 클러스터링을 수행할 수도 있다. 두 가지 유형의 벡터를 동시에 고려할 수 있도록 지원하는 이유는 프로세스 인스턴스 간에 중복성이 낮은 경우에는 액티비티 벡터가 유사성 판정에 주요한 역할을 하며, 중복성이 높은 경우에는 트랜지션 벡터가 순서까지 고려하여 정밀한 유사성 판정 역할을 하기 때문이다.

AHC 알고리즘은 가장 유사한 프로세스 인스턴스 벡터들부터 우선적으로 군집화하면서 점차 큰 클러스터들을 생성해 나가는 과정으로, 그 결과를 계층적 구조로 표현할 수 있기 때문에 사용자가 군집화 과정을 이해가 쉬우며, 실험 결과로부터 적절한 군집개수를 판별하는 것이 용이하다[12]. 알고리즘의 개략적인 유사 코드는 <Figure 2>에 나타나 있다.

반면에, K means 알고리즘은 클러스터의 개수를 미리 결정한 후 그 개수만큼의 초기 프로세스 인스턴스 벡터(seed)를 선택한 후 각 벡터를 가장 유사한 클러스터로 할당시키면서 클러스터링을 진행하는 방법이다[12]. 결과 클러스터 내의 어떠한 프로세스 벡터들이 더 유사한 관계를 가지는지, 최종 클러스터의 개수를 변경하면 어떻게 클러스터를 생성할지를 판단하기가 어려운 단점이 있다. 알고리즘의 개략적인 유사 코드는 <Figure 3>에 나타나 있다.

## 4. PROCL : 프로세스 로그 클러스터링 시스템

본 연구에서는 프로세스 마이닝 오픈 프레임워크인 ProM에서 활용할 수 있도록 프로세스 클러스터링 기법을 구현하였다. ProM은 자바로 구현된 오픈 프레임워크로 Petri-net, EPC(Event driven Process Chain), BPEL(Business Process Execution Language) 등의 다양한 프로세스 정의 언어를 읽어들여 프로세스 모델로 분석할 수 있는 변환기를 제공하며, SAP R/3, IBM Websphere, Woflan,

```
L = makeLogVectors() ; C = L ;
S = compute_similarity(L) ;
while n(C) < k do
  (u, v) = find_nearest_pair(L) ;
  w = merge(u, v) ;
  update_cluster(L, u, v, w) ;
  update_similarity(S, w) ;
end while
```

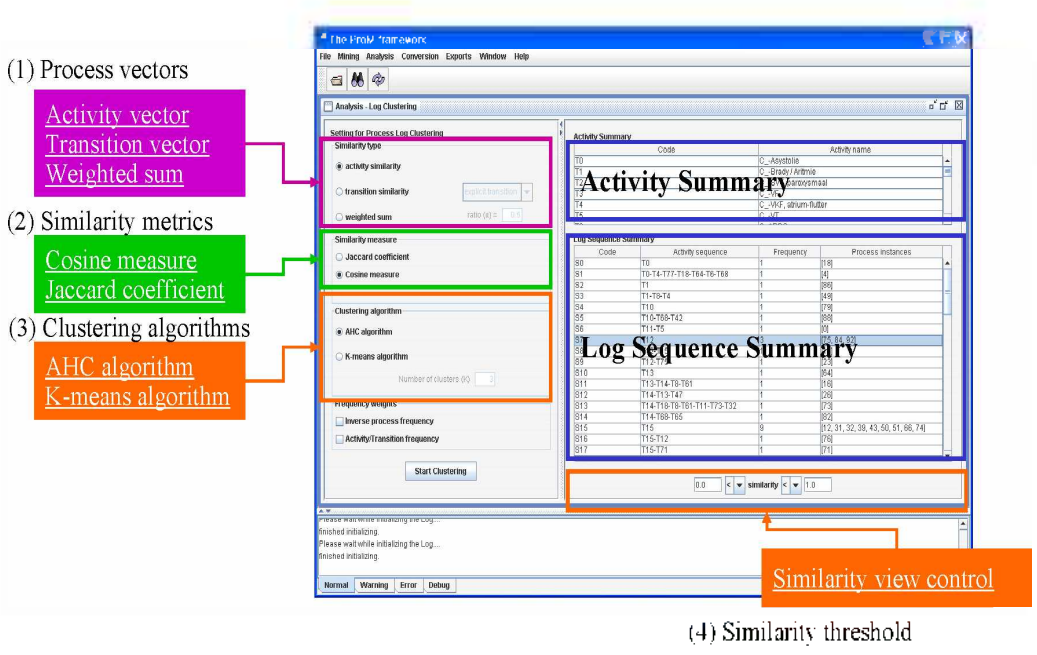
〈Figure 2〉 AHP algorithm

```
C = choose_centroid(L, k) ;
while delta(C, C') < threshold do
  S = compute_similarity(L, C)
  (u, v) = select_max(S)
  C = merge(u, v)
  S = update_similarity(u, C)
end while
```

〈Figure 3〉 K-means algorithm

ARIS, Protos, Flower 등의 ERP 시스템, BPM 시스템, BPR 시스템에서 발생된 프로세스 실행 이벤트 로그들을 변환하여 여러 가지 마이닝 알고리즘으로 프로세스 모델을 추출할 수 있는 기능을 제공한다. http://prom.sourceforge.net/에서 다운받을 수 있으며, 매 포판에는 본 연구에 제시된 로그 클러스터링의 이전 버전이 포함되어 있다.

본 연구의 구현 시스템인 PROCLE은 ProM에서 읽어들인 프로세스 인스턴스들로부터 3장에서 설명한 세 가지 프로세스 로그 벡터들을 추출한 후, 유사성 측도를 지정하고, 원하는 클러스터링 알고리즘을 통하여 프로세스 로그들을 분류할 수 있는 기능을 제공한다. 이 클러스터링의 결과는 기존의 ProM 시스템과 연계되어, 사용자가 원하는 로그 클러스터를 선택하여 기존에 개발되어 있는 프로세스 마이닝 알고리즘을 실행할 수 있도록 연결하였다. AHP 알고리즘의 장우이는, 트리 구조인 덴드로그램(dendrogram)으로 로그 클러스터들을 보여준 후, 원하는 수준의 로그 클러스터에 대하여 alpha mining, heuristic mining, social mining 등의 다양한 프로세스 마이닝 기법이 적용 가능하다. Alpha mining이나 heuristic mining 알고리즘은 프로세스 로그에서 적절한 프로세스 모델을 역으로 추출하는 기법이며[15], social mining 알고리즘은 프로세스 로그에서 실행 중에 참여한 조직 및 작업자들의 상호연관관계를 분석하는 기법이다[14]. 즉, 프로세스 클러스터링 분석 기능은 방대한 프로세스 인스턴스 정보를 프로세스 유사성에 기반으로 분류함으로써, 사용자가 원하는 적정한 수준의 프로세스 모델을 추출하는 목적으로 활용될 수 있다.




〈Figure 4〉 Setting for process log clustering

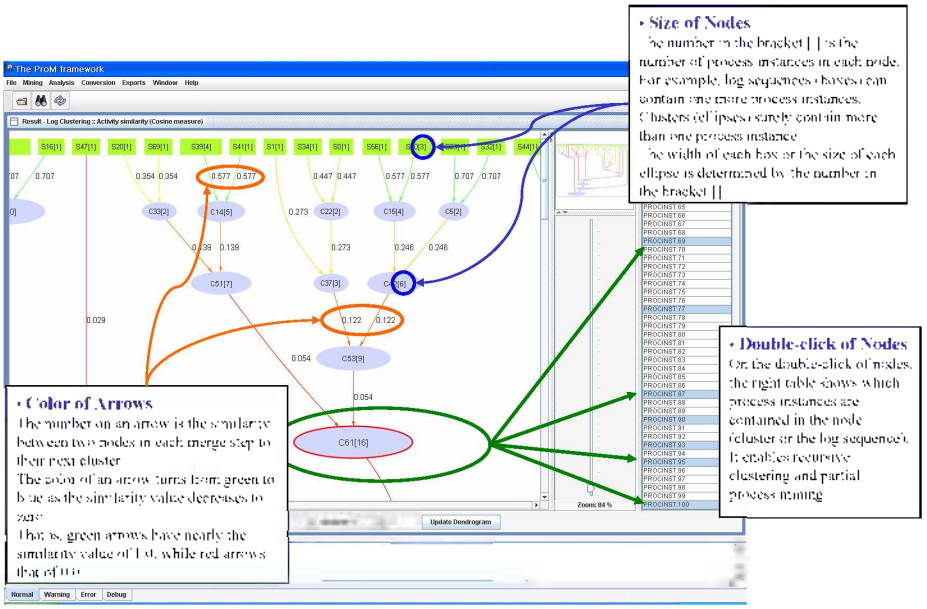
## 4.1 클러스터링 고려요소 결정

먼저, PROCL은 프로세스 로그로부터 전체 액티비티를 추출하여 인코딩하고, 이를 기반으로 각 프로세스 인스턴스를 액티비티 리스트로 인코딩한다. 동일한 액티비티 순서를 가지는 인스턴스의 경우, 동일한 프로세스 인스턴스로 간주하며, <Figure 4>의 "Log Sequence Summary" 부분이 프로세스 인스턴스의 인코딩 결과를 보여준다. 이 프로세스 인스턴스를 바탕으로 로그 클러스터링을 수행한다.

<Figure 4>는 프로세스 클러스터링을 수행하기 위한 설정 화면이다. (1) Process Vectors, (2) Similarity metrics, (3) Clustering algorithms에서는 제 3장에서 설명한 각 수행 단계에서의 고려사항을 설정할 수 있다. 추가로 (4)에서는 클러스터링의 수준을 선택할 수 있도록 Similarity threshold(유사성 한계치)를 지원한다. 실제 프로세스 저장소에 존재하는 프로세스 로그의 개수는 상당히 방대하기 때문에, AHP 알고리즘의 경우 하나의 트리 구조로 제시하는 것은 가독성이 떨어진다. 이를 방지하기 위하여 사용자가 원하는 수준의 유사성 한계치를 지정함으로써, 상위 수준의 로그 클러스터만을 보여주어 상위 클러스터간의 유사성을 관찰하거나, 낮은 수준의 로그 클러스터만을 보여주어 하위 클러스터를 구성하는 더 낮은 수준의 클러스터 또는 프로세스 인스턴스 구성을 관찰할 수 있다. 이처럼 사용자가 모니터링 하고자 하는 유사성의 범위를 지정하는 기능을 제공한다.

## 4.2 클러스터링 실행 결과

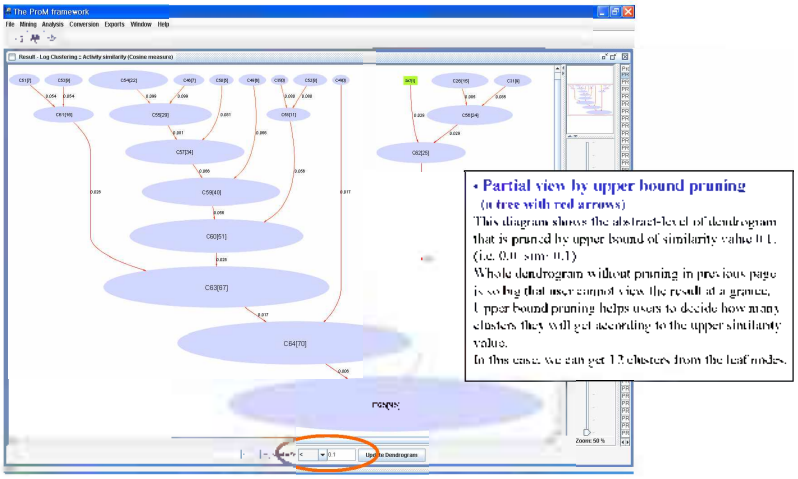
본 절에서는 프로세스 로그 클러스터링 실



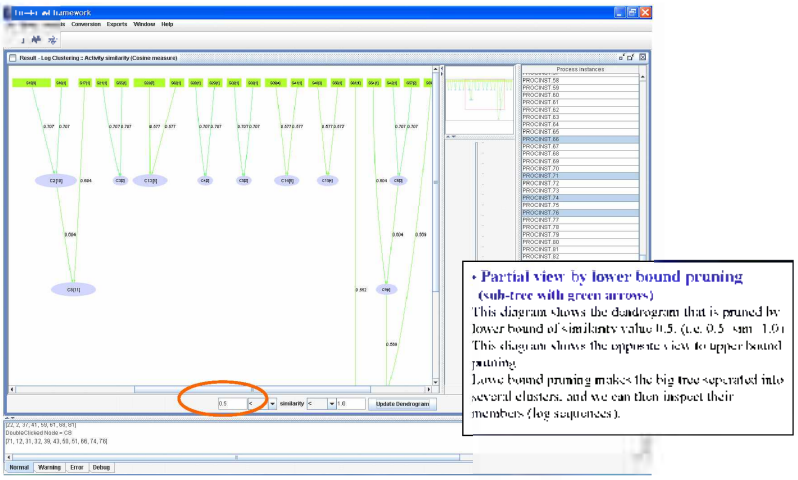

〈Figure 5〉 Result of hierarchical process log clustering

행 결과를 설명한다. 먼저, <Figure 5>은 계층적 클러스터링의 결과 화면인 덴드로그램을 보여준다. 이 덴드로그램은 클러스터들이 어떤 단계로 군집을 형성하는지 보여주고 있으며, 각 클러스터를 선택하면 화면 오른쪽의 프로세스 인스턴스 목록을 통하여 어떤 프로세스 인스턴스들이 그 클러스터에 속해있는지를 확인한 후, ProM 프레임워크에서 지원하는 Heuristic mining, Alpha mining, Social mining 등의 프로세스 마이닝 알고리즘을 재적용할 수 있도록 지원한다.

트리 구조로 표현된 덴드로그램에는 두 가지 노드가 포함되어 있다. 사각형은 각 프로세스 인스턴스로부터 추출된 동일한 프로세




(a) upper bound pruning




(b) lower bound pruning

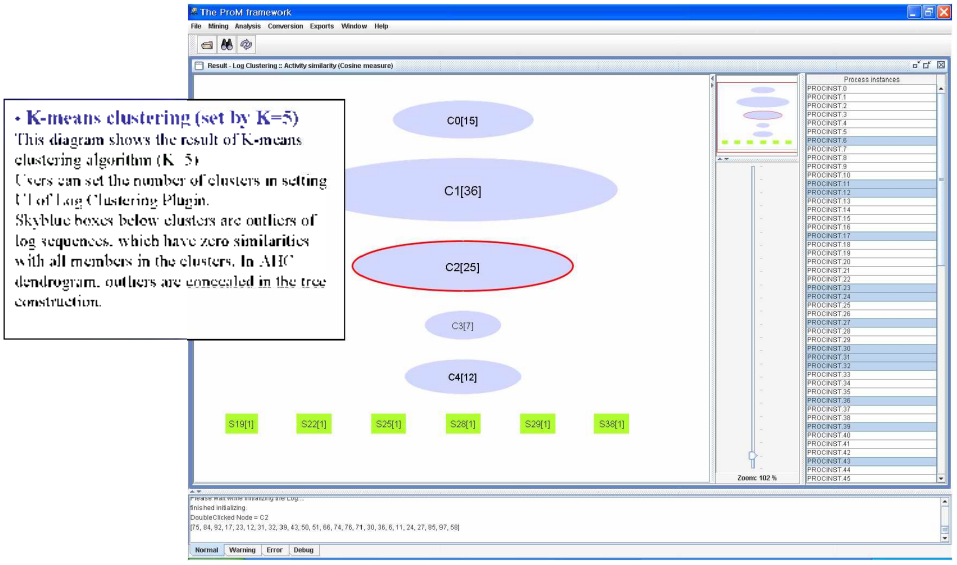
〈Figure 6〉 Pruning results of hierarchical process log clustering

스 로그 벡터이며, 타원은 여러 개의 프로세스 벡터로 구성된 로그 클러스터이다. 로그 클러스터는 포함된 프로세스 인스턴스의 개수에 따라 타원의 크기가 결정되며, 그 개수는 대괄호 [] 안에 표시되어 있다. 또한, 프로세스 로그나 프로세스 클러스터를 더블 클릭을 하면, 오른쪽 프로세스 인스턴스 리스트에서 포함된 인스턴스가 표시된다. 실정을 변경하여 로그 클러스터링을 재적용하기 위하여, 또는 기존의 프로세스 마이닝을 실행하기 위하여 입력 프로세스로 사용될 수 있다.

군집화되는 과정에서 두 하위 군집의 유사성 값은 각 아크(arc)에 표현된 0에서 1사이의 실수값으로 주어진다. 실수값이 1에 가까운 경우는 매우 유사한 것을 의미하고 녹색에 가깝게 표시되며, 실수값이 0에 가까운 경우는 유사성 정도가 낮은 것으로 빨간색에 가깝게 표시된다.

<Figure 5>의 프로세스 클러스터링 결과는 방대한 노드 수로 인하여 덴드로그램을 한눈에 쉽게 판별할 수가 없다. 그래서 4.1절에 설명한 바와 같이 유사성 한계치를 지정하여 하위 그래프를 생성하여 원하는 수준의 트리를 생성할 수 있다. <Figure 6>의 (a)와 (b)는 각각 0.1 이하의 유사성 구간과, 0.5이상의 유사성 범위로 제한한 덴드로그램을 보여주고 있다.

한편, <Figure 7>은 K means 알고리즘을 이용한 프로세스 클러스터링 결과를 보여준다. K means 알고리즘은 클러스터 형성 과정을 시각적으로 보여주지는 못하지만, 클러스터의 크기를 통하여 클러스터에 속한 로그의 개수를 보여준다. 또한, 아래의 개별적으로 표현된 프로세스 로그들은 외곽값(outlier)들로서(그림의 경우 6개의 outlier가 발생하였음) 다른 클러스터와의 유사성이 0으로 다른 클러스터에 포함되지 못한 로그들을 보여준다. 데이터의 분석 결과 이러한 외곽값들은 단일 액티비티로 구성된 로그가 대부분이었다.




〈Figure 7〉 Pruning results of process log clustering by K-means algorithm

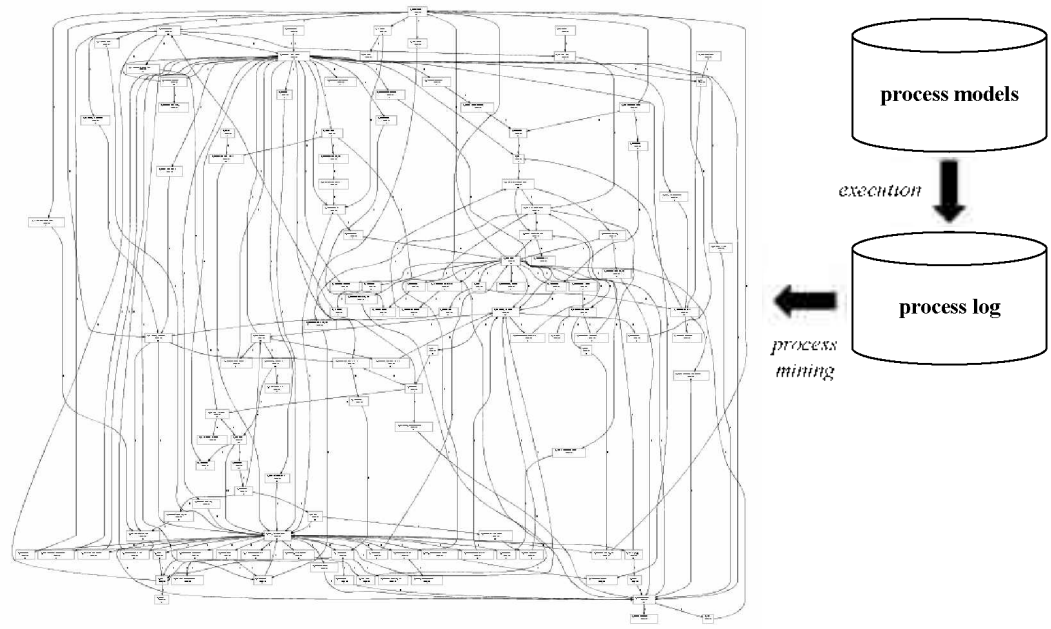
### 4.3 클러스터링을 활용한 프로세스 마이닝

기존의 프로세스 마이닝 연구들은 프로세스 로그 저장소 내에 저장된 프로세스 인스턴스를 대상으로 정보를 추출한다. 그러나 로그의 규모가 크거나, 다양한 프로세스 모델이 섞여있는 경우도 일괄적으로 처리함으로써 프로세스 마이닝 결과가 이해할 수 없는 경우들이 많았다. <Figure 8>은 한 기업의 워크플로우 시스템에서 생성된 프로세스 로그를 대상으로 프로세스 마이닝한 결과이다. 이는 사람이 이해하기 힘든 정도로 스파게티 같은 프로세스 그래프를 생성하였다.

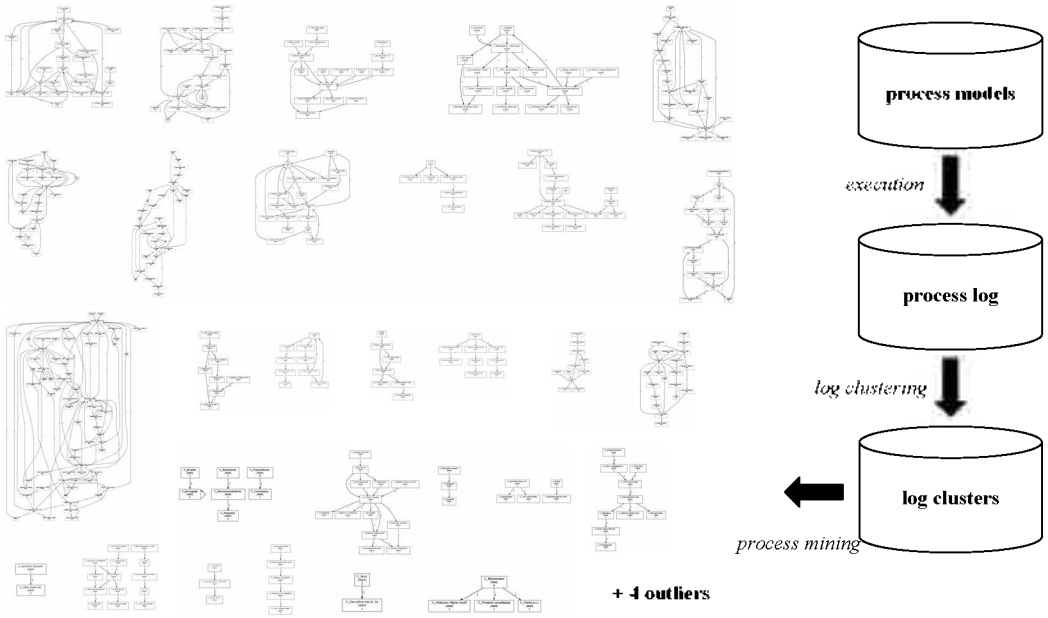
한편, 본 연구에서 제안된 프로세스 클러스터링은 프로세스 저장소 내의 로그들을 유사성에 기반하여 선처리 함으로써 프로세스 로그들을 분류하는 작업을 수행한다. 그 결과는 프로세스 클러스터들로 분류되어 기존의 여러 프로세스 마이닝 기법들을 적용할 수 있도록 지원한다. <Figure 9>는 <Figure 8>과 동일한 프로세스 로그를 먼저 클러스터링을 적용한 후, 프로세스 마이닝을 수행한 결과이다. 로그 클러스터링을 수행하여 39개의 프로세스 클러스터를 생성하였으며(외곽값의 개수는 4개), 각 클러스터를 프로세스 마이닝한 결과인 프로세스 그래프들을 보여준다. <Figure 8>과 비교하여 간단하고 이해할 수 있는 수준으로 분할된 프로세스 그래프를 생성할 수 있었다.

## 6. 결 론

프로세스 마이닝은 최근 비즈니스 프로세스 분석을 위하여 활발히 수행되는 분야로




〈Figure 8〉 A complicated process model extracted by general process mining




〈Figure 1〉 Process models extracted by process mining accompanied with log clustering

이러 가지 알고리즘 및 기법들이 제시되고 있다. 프로세스 로그 클러스터링이 프로세스 마이닝에서 갖는 의미는 다음과 같다. 기존의 프로세스 마이닝은 프로세스 로그 저장소 내의 전체 로그를 대상으로 적용되었기 때문에, 로그가 방대하거나 다양한 프로세스 모델들이 섞이있는 경우에 가독성이 낮은 무의미한 프로세스 그래프를 추출하는 경우가 많았다. 본 연구에서 제시된 프로세스 로그 클러스터링 과정은 프로세스 마이닝 이전에 전처리로 활용됨으로써 원하는 수준의 프로세스 모델들을 추출하는 방안을 제시한다. 앞으로는 이러 가지 프로세스 유사성 측정 방법과 개선된 클러스터링 알고리즘을 적용함으로써 로그 분류 기능을 개선할 수 있을 것이며, 본 논문에서 제시된 방법은 프로세스 실행 결과 분석 및 검증을 위하여 계속 개발중인 이타 프로세스 마이닝 알고리즘들과 결합하여 널리 사용될 수 있을 것으로 예상된다.

## 참 고 문 헌

[1] Bae, J, Kwon, B., Jung, J. Y., and Kang, S.-H., "Workflow Collaboration Design using Similarity Measures among Process Definitions," Review of Korean Society for Internet Information, Vol. 6, No. 1, 2005, pp. 52 61.

[2] Cardoso, J., "How to Measure the Control-flow Complexity of Web processes and Workflows," In : Fischer, L.(ed.) :

Workflow Handbook 2005, WfMC, Lighthouse Point, FL, 2005, pp. 199-212.

[3] de Medeiros, A. K. Alves, A. J. M. M. Weijters, and W. M. P. van der Aalst, "Genetic Process Mining : An Experimental Evaluation," Data Mining and Knowledge Discovery, Vol. 14, No. 2, 2007, pp. 245-304.

[4] Hur, W., Bae, H., and Kang, S.-H., "Customizable Workflow Monitoring," Concurrent Engineering-Research and Applications, Vol. 11, No. 4, 2003, pp. 313-326.

[5] Jansen-Vullers, M. H., van der Aalst, W. M. P., and Rosemann, M., "Mining Configurable Enterprise Information Systems," Data and Knowledge Engineering, Vol. 56, No. 3, 2006, pp. 195-244.

[6] Jung, J., Hur, W., Kang, S.-H., and Kim, H., "Business Process Choreography for B2B Collaboration," IEEE Internet Computing, Vol. 8, No. 1, 2004, pp. 37-45.

[7] Kim, H., Jung, J.-Y., and Kang, S.-H., "Extensible Collaborative Process Composition using Workflow Inheritance," IE Interfaces, Vol. 16, 2003, pp. 49-54.

[8] Kim, Y., Kang, S. H., Kim, D., Bae, J., and Ju, K. J., "WW-Flow : Web-Based Workflow Management with Runtime Encapsulation", IEEE Internet Computing, Vol. 4, No. 3, 2000, pp. 55-64.

[9] Reijers, H. A. and Vanderfeesten, I. T. P., "Cohesion and Coupling Metrics for Workflow Process Design," Proc. 2nd Int. Conf. on Business Process Management, 2004, pp. 290-305.

[10] Rozinat, A. and W. M. P. van der Aalst, "Decision Mining in ProM," Proc. 4th Int. Conf. on Business Process Management, 2006, pp. 420-425.

[11] Salton, G. and McGill, M. J., Introduction to Modern Information Retrieval. McGraw-Hill, 1983.

[12] Tan, P. N., Steinbach, M., and Kumar, V., Introduction to Data Mining, Addison Wesley 2005.

[13] van der Aalst, W. M. P., and Basten, T., "Inheritance of workflows : an approach to tackling problems related to change," Theoretical Computer Science, Vol. 270, No. 1, 2002, pp. 125-203.

[14] van der Aalst, W. M. P., H. A. Reijers, and M. Song, "Discovering Social Networks from Event Logs," Computer Supported Cooperative Work, Vol. 14, No. 6, 2005, pp. 549-593.

[15] van der Aalst, W. M. P., et al., "Business process mining : An industrial application," Information Systems, Vol. 32 , No. 5, 2007, pp. 713-732.

[16] van der Aalst, W. M. P., et al., "ProM 4.0 : Comprehensive Support for Real Process Analysis," Proc. 28th Int'l Conf. on Applications and Theory of Petri Nets and Other Models of Concurrency (ICATPN 2007), LNCS, Vol. 4546, 2007, pp. 484-494.

[17] van der Aalst, W. M. P., "Trends in Business Process Analysis : From Verification to Process Mining," Proc. 9th Int'l Conf. on Enterprise Information Systems (ICEIS 2007), 2007, pp. 12-22.

## 저 자 소 개

| | |
|---|---|
| 정제윤 | (E mail : jyjung@khu.ac.kr) |
| 1999년 | 서울대학교 산업공학과 (학사) |
| 2001년 | 서울대학교 산업공학과 (석사) |
| 2005년 | 서울대학교 산업공학과 (박사) |
| 2005년 ~ 2006년 | 네덜란드 아인트호벤공대 초빙연구원 |
| 2006년 ~ 2007년 | 서울대학교 유비쿼터스 컴퓨팅 원천기술 개발지원 센터 |
| 2007년 ~ 현재 | 성희대학교 산업공학과 전임강사 |
| 관심분야 | 비즈니스 프로세스 관리(BPM), 유비쿼터스 서비스 컴퓨팅, 인터넷 비즈니스 |