Journal of the Korean Institute of Industrial Engineers
Vol. 33, No. 1, pp. 99-109, March 2007.

프로세스 유사성을 이용한 워크플로우 클러스터링

정재윤[1] · 배준수[2†] · 강석호[3]

[1] 서울대학교 자동화시스템공동연구소 / [2] 전북대학교 산업정보시스템공학과
[3] 서울대학교 산업공학과

Workflow Clustering Methodology Using Structural Similarity Metrics

Jae-Yoon Jung[1] · Joonsoo Bae[2] · Suk-Ho Kang[3]

[1] Automation and Systems Research Institute, Seoul National University
[2] Department of Industrial and Information Systems Engineering, Chonbuk National University
[3] Department of Industrial Engineering, Seoul National University

To realize process-driven management, so many companies have been launching business process management systems. Business process is collection of standardized and structured tasks inducing value creation of a company. Moreover, it is recognized as one of significant intangible business assets to achieve competitive advantages. This research introduces a novel approach of workflow process analysis, which has more and more significance as process-aware information systems are spreading widely into a lot of companies. In this paper, a methodology of workflow clustering based on process similarity has been proposed. The purpose of workflow clustering is to analyze accumulated process definitions in order to assist design of new processes and improvement of existing ones. The proposed methodology exploits measures of structural similarity of workflow processes.The methodology has been experimented with synthetic process models for illustrating the implication of workflow clustering.

Keywords: Business Process Analysis, Workflow Clustering, Process Similarity

1. 서 론

비즈니스 프로세스 관리는 RTE(Real-Time Enterprise), BSC (Balanced Scored Card), KM(Knowledge Management) 등의 다양한 경영혁신이론을 프로세스 중심의 정보 시스템을 기반으로 적용하고 평가하기 위한 목적으로 최근에 활발히 도입되고 있다. 비즈니스 프로세스 관리 시스템은 두 가지 핵심 기술로 구성되는데, 하나는 특정 영역에 특화된 다양한 업무 지원 시스템(예, ERP, CRM, SCM, groupware 등)을 연계하기 위한 애플리케이션 통합(EAI : Enterprise Application Integration) 기술이며, 다른 하나는 조직 모델을 바탕으로 업무 전달을 자동화하고 통제하기 위한 워크플로우(workflow) 기술이다. 특히, 워크플로우 기술은 프로세스 자동화 기술의 원천이며, 전략적 기업 경영을 위한 프로세스 중심의 경영혁신을 실현하는 토대로서 비즈니스 프로세스의 실시간 모니터링 및 성능 평가 기반을 제공한다. 초기 워크플로우 기술은 프로세스 자동화 및 실행 관리로부터 출발하여(Kim *et al*., 2000; Jung *et al*., 2004), EIP(Enterprise Information Portal), BAM(Business Activity Monitoring), RTE 등 실시간 정보공유 및 모니터링을 지원하는 연구들을 거쳐(Hur *et al*., 2003), 최근에는 워크플로우의 실행 결과를 분석하는 워크플로우 마이닝(workflow mining)이나(Aalst and Weijters, 2004; Jansen-Vullers *et al*., 2005), 업무 할당을 분

이 논문은 2005년도 한국학술진흥재단의 지원에 의하여 연구되었음(KRF-2005-214-D00192).
† 연락저자 : 배준수, 561-756 전북 전주시 덕진구 덕진동 전북대학교 산업정보시스템공학과, Tel : 063-270-2332, Fax : 063-270-2333, E-mail : jsbae@chonbuk.ac.kr
2005년 12월 접수; 2006년 09월 수정본 접수; 2006년 10월 게재 확정.

석하는 작업부하 분산(Simitsis *et al*., 2005; Ha *et al*., 2006) 등 비즈니스 프로세스의 분석 및 개선을 위한 방법론을 제시하고 있다. 특히, 비즈니스 프로세스의 유형과 분류에 관한 연구는 프로세스 인식 가능한 시스템이 점점 증가하고 있으며 많은 기업들이 프로세스 기반 정보 시스템을 도입함에 따라서 그 중요성이 증가하고 있다. 현재까지의 프로세스의 유형과 분류에 관한 연구는 경영학적 입장에서 보편적 업무 성격에 근거한 참조 프로세스를 제공하기 위하여 시도되거나(Malone *et al*., 2003), 기업간 전자거래를 위하여 필요한 표준 프로세스를 제공하기 위하여(RosettaNet, 2001) 사용되었다. 그러나 이들은 기업 자신의 프로세스 자산을 관리하기 위한 방안은 아니며, 현재와 같이 다양한 프로세스 인식 시스템을 통하여 정의되고 실행되고 있는 프로세스의 분석은 제공하지 못하고 있다.

본 연구에서는 프로세스 기반 정보시스템에서 누적되고 있는 프로세스 모델을 분석하기 위한 한 가지 방안으로서 워크플로우 클러스터링 방법론을 제안하였다. 워크플로우 클러스터링이란 프로세스 저장소에 존재하는 프로세스 모델 간 구조적 유사성을 측정하여 유사한 프로세스 군집을 찾아내는 과정이라고 정의할 수 있다. 유사한 프로세스 모델을 식별하여 군집화하는 작업은 프로세스 설계자가 새로운 자동화 프로세스를 설계하는 데 참조하기 위한 프로세스를 추천해 줄 수 있으며, 비즈니스 프로세스 모델의 유형별 패턴을 분석함으로써 대상 프로세스를 개선하기 위하여 사용될 수 있다. 본 연구에서는 프로세스 설계 지원을 위한 워크플로우 추천 시스템을 개발하기 위한 목적으로 워크플로우 클러스터링 연구를 수행하였다. 본 연구에서는 워크플로우 프로세스를 군집화하기 위하여 프로세스의 구조적 유사성 척도를 사용하게 되며, 이 척도는 그래프 유사성 척도를 확장하여, 업무 수행의 의존 관계를 표현하고 있는 프로세스 그래프의 유사성을 반영하기 위하여 개발되었다.

제안된 워크플로우 클러스터링의 의미와 효과를 설명하기 위하여, 본 연구에서는 프로세스 모델을 모의로 생성하여 실험을 수행하였다. 워크플로우 클러스터링은 추후 프로세스들이 다양한 정보 시스템에서 점점 더 많이 정의되고 축적됨에 따라 이를 자동으로 분류하고 분석하기 위한 한 가지 방안으로서 제시되었으며, 프로세스 중심 시스템 환경에서 프로세스 자산을 체계적으로 분석함으로써 신규 프로세스 설계나 기존 프로세스 분석 등에 다양하게 적용될 수 있을 것이다.

본 논문은 다음과 같이 구성된다. 먼저 제 2장에서 배경 연구에 관하여 설명한다. 제 3장에서 워크플로우 프로세스의 구조를 표현하는 프로세스 벡터를 생성하는 방법과 프로세스의 유사성 척도를 제시한다. 제 4장에서는 워크플로우 클러스터링의 프레임워크를 제시하고, 계층적 클러스터링 알고리즘에 관하여 설명한다. 제 5장에서는 모의 데이터를 사용한 워크플로우 클러스터링에 관한 실험 결과를 설명한 후, 마지막으로 제 6장에서 본 연구의 결론을 기술하였다.

2. 배경 연구

많은 기업으로 비즈니스 프로세스 관리 시스템이 확산되고 있지만, 복잡한 프로세스를 설계하고 분석하는 방법에 관한 연구는 여전히 미비한 상태이다. 다양한 목적의 비즈니스 프로세스의 유형과 분류에 관한 오랜 연구 결과물로서 MIT의 프로세스 핸드북 작업이 있다(Malone *et al*., 2003). 여기에는 5000가지 이상의 영역별, 기능별 프로세스 저장소를 구축하고 있으며, 구매, SCM, 마케팅, 영업, 정보시스템, 재무, 공학 등의 영역과 기능에 따라 프로세스들을 분류하고 각 특성과 업무를 기술하고 있다. 이 연구는 영역과 목적 위주로 프로세스 저장소를 구축한 대표적 연구로서, 프로세스 분류를 통하여 프로세스 설계를 위한 참조 모델을 제공한다는 점에서 본 연구와 동일한 목적을 가지지만, 조직 자신이 보유한 프로세스를 기반으로 하는 것이 아니라, 모든 기업들이 참조할 수 있는 보편적이고 표준화된 프로세스 모델을 제공한다는 점에서 본 연구와 대조적이다. 한편, 조직 자신의 프로세스 분석에 관한 대표적인 연구로는 프로세스 마이닝을 들 수 있다(Aalst and Weijters, 2004; Schimm, 2004; Jansen-Vullers *et al*., 2005). 프로세스 마이닝은 프로세스 저장소의 실행 결과 및 이벤트에 관한 로그를 분석함으로써, 업무의 의존 관계를 표현할 수 있는 프로세스 모델을 유도하거나, 업무의 상관관계, 작업자의 업무 전달 관계 등의 프로세스 수행상의 특징을 분석하는 연구이다. 이 연구는 확산되고 있는 프로세스 기반 정보시스템을 기반으로 분석한다는 점에서 본 연구와 유사한 배경을 가지지만, 이 연구는 프로세스 실행 로그를 대상으로 분석하는 반면에, 본 연구는 프로세스 모델을 대상으로 비교 분석한다는 점에서 차이가 있다.

프로세스 모델의 구조적 분석을 통하여 프로세스 설계 및 개선에 활용하는 연구로는 프로세스 상속, 프로세스 비교 척도, 프로세스 평가 척도에 관한 연구들이 있다. Aalst and Basten (2002)와 Kim *et al*.(2003) 등의 프로세스 상속에 관한 연구는 프로세스 간의 종속 관계를 분석함으로써 프로세스의 동적 확장 가능성을 탐색하기 위한 연구이다. 이들 연구는 기업간 프로세스 관리로 프로세스 모델을 확장하거나 변화 관리를 위해 프로세스의 동적 변화를 지원하기 위한 목적으로 제안되었다. 한편, Bae *et al*.(2005, 2006)에 의해 진행된 프로세스 비교 척도에 관한 연구는 신규 프로세스를 공동 설계할 때, 다수의 전문가가 설계한 프로세스 모델을 비교하여 하나의 통합 프로세스 모델을 도출하기 위하여 수행되었다. 또한, 최근에 프로세스 모델의 안정성과 견고함을 평가하기 위한 프로세스 평가 척도에 관한 연구들이 진행되고 있다. Cardoso(2005)의 프로세스 모델의 복잡도 척도는 프로세스 모델의 분기와 병합에 대한 복잡도를 평가하여 실행 오류를 줄이기 위한 척도로서 제안되었으며, Reijers and Vanderfeesten(2004)의 프로세스 모델의 응집성 및 결합성 척도는 제품 정보와 같이 프로세스 모델에 포함된 정보의 전달 방식을 평가함으로써 응집성과 결합성이라

는 두 척도에 대한 균형적인 프로세스 모델을 도출하기 위하여 사용되었다. 이처럼 프로세스 상속, 비교 척도, 평가 척도를 비롯한 프로세스 모델 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 모델 비교에 의한 클러스터링은 프로세스의 구조적 관점에서 프로세스 모델을 분석한다는 점에서 프로세스 상속 및 평가 척도와 같은 성격을 지니고 있지만, 프로세스 설계를 추천하기 위한 목적이라는 점에서 차이를 보이고 있다. 또한, 그룹 설계를 지원하기 위한 프로세스 비교 척도(Bae *et al*., 2005, 2006)의 연구는 워크플로우의 다양한 요소들을 사용하고 있으나, 본 연구는 구조적 관점에 집중하여 유사한 프로세스를 추천함으로써 프로세스 신규 설계를 지원하는 것을 목적으로 수행되었다.

프로세스의 구조 분석을 위하여 사용되는 유용한 방법 중 하나는 그래프 구조에 기반한 프로세스의 분석이다. 특히 본 연구에서는 프로세스 간의 유사성 척도를 개발하기 위하여 그래프 구조를 사용하였다. 그래프에 기반한 유사성 척도는 다양한 분야에서 적용되는 데이터 구조 분석 기법으로, 문자나 형상 인식, 웹이나 XML 문서 분석, 스키마 분석 등 패턴 매칭 및 기계 인식에서 유용하게 활용되고 있다(Bunke and Shearer, 1998; Hammouda and Kamel, 2004). 그래프 구조에 관한 비교 연구는 전통적으로 그래프 동형성(graph isomorphism) 분석을 바탕으로 이루어졌으며(Corneil and Gotlieb, 1970; Ulman *et al*., 1976), 대표적인 그래프 유사성 척도로는 노드와 에지의 추가, 삭제, 수정 과정의 비용함수를 이용하는 그래프 편집 거리(GED : graph edit distance) 방법(Messmer and Bunke, 1998; Bunke, 1999)과, 공유하는 하위 그래프를 측정하는 최대 공통 하위그래프(MCS : maximal common subgraph) 방법(Bunke and Kandel, 2000; Fernandez and Valiente, 2001)이 널리 활용되고 있다(Zamir and Etzioni, 1998; Zhang and Shasha, 1989). 이러한 척도는 문서의 유사성 분석 및 클러스터링에서도 사용되고 있으며, 문서의 검색엔진이나, 문서 관리 및 지식 저장소 구축 등에 적용되고 있다(Bunke and Shearer, 1998; Hammouda and Kamel, 2004). 본 연구의 대상인 프로세스와 마찬가지로 그래프 구조를 가지는 XML 문서에 대한 구조적 유사성 척도를 이용한 연구도 진행되고 있다(Lian *et al*., 2004). 본 연구는 이처럼 다양한 분야에서 시도 중인 그래프 구조를 이용한 유사성 측정 및 클러스터링에 관한 연구 중 하나라고 간주될 수 있다.

3. 프로세스의 구조적 유사성

워크플로우 프로세스는 업무 흐름과 통제를 컴퓨터를 이용하여 자동으로 실행하기 위해 설계된 비즈니스 프로세스이며(WfMC, 1999), WfMC(Workflow Management Coalition)에서 제안된 XPDL(XML Process Definition Language)와 같이(XPDL, 2002) 워크플로우 프로세스들은 특수한 형태의 방향성 그래프(labelled directed graph)를 이용하여 설계된다.

워크플로우간의 유사성은 크게 두 가지 측면에서 논의될 수 있다. 하나는 의미론적인 유사성(semantic similarity)이며, 다른 하나는 구조적인 유사성(structural similarity)이다. 의미론적 유사성은 비즈니스 프로세스의 목적과 업무 수행에 관련된 구성요소의 근사한 정도를 측정할 수 있다. 워크플로우의 구성요소로는 프로세스와 액티비티를 비롯하여, 이를 수행하는 데 사용되는 입출력 데이터와 문서, 애플리케이션, 전이조건 및 업무담당자 등 다양한 요소들이 존재하며, Bae *et al*.(2005, 2006)의 연구에서와 같이 이러한 다양한 요소들을 비교함으로써 워크플로우간의 유사성을 측정할 수 있다. 나아가 의미론적 유사성을 측정하기 위하여 온톨로지, 시맨틱 웹, 작업분해구조, 지식맵 등의 도구들이 사용될 수 있다.

Table 1. Categories of workflow similarity

	의미론적 유사성	구조적 유사성
비교 대상	- 워크플로우 사용목적 - 액티비티의 업무내역 - 관련정보(입출력자료) 및 활용 애플리케이션 - 참여자 및 조직 모델	- 동일 액티비티의 등장 - 액티비티들의 순차적 관계 - 분기 및 순환 관계
비교 방법	온톨로지, 시맨틱 웹, 작업분해구조(WBS), 지식맵 등의 도구가 활용 될 수 있음	프로세스 모델에 따라 방향성 그래프 기반, 블록 구조 기반, Petri-net 기반 방법 등으로 분류 가능

반면에 구조적 유사성은 워크플로우를 자동으로 실행하는 데 필요한 업무의 순서와 의존관계를 바탕으로 수행될 수 있다. 워크플로우 프로세스를 정의하는 방법은 크게 방향성 그래프(directed graph) 방법과 블록 구조(block structured) 방법으로 분류되는데, 방향성 그래프에 기반한 방법은 사용자의 설계 및 이해 용이도가 뛰어나고 표현 능력이 풍부하여, 워크플로우나 비즈니스 프로세스 설계 시스템에서 많이 사용된다(Shapiro, 2002). 이에 비하여 블록 구조 방법은 흐름 통제가 명확하고 자동화하기가 용이하여 최근 전자거래를 위한 비즈니스 프로세스 표준 언어(BPEL, BPML, WSCI 등)에서 많이 사용된다. 블록 구조의 프로세스 정의는 방향성 그래프로 변환이 가능하므로, 본 연구에서는 프로세스 구조적 분석 대상으로 방향성 그래프 기반의 유사성 척도를 개발하였다.

방향성 그래프 기반의 프로세스 유사성 분석에는 기존의 그래프 유사성에 관한 연구가 활용될 수 있다. 그 중 대표적인 그래프 거리 척도인 그래프 편집 거리(GED)와 최대 공통 하위그래프(MCS)를 <Figure 1>의 간단한 예를 통하여 살펴보자. P_0와 P_1은 두 그래프 거리 척도에서 매우 유사하게 측정된다. 하지만, P_1과 P_2는 그래프 편집 거리 척도에서는 매우 유사하지

만, 최대 공통 하위그래프 척도에서는 전혀 유사하지 않다. 워크플로우 프로세스의 측면에서는 대체 가능 업무인 액티비티 2와 액티비티 4가 대체되었다고 본다면 매우 유사한 프로세스로 간주될 수 있다. 한편, P_3과 같이 분기가 포함된 프로세스의 경우, 유사성이나 거리 척도는 분기의 해석에 상당히 의존적이다. 아래 절에서는 워크플로우 클러스터링에 사용되는 프로세스 구조의 표현 방법과 이를 이용한 프로세스의 구조적 유사성 척도에 관하여 설명한다.

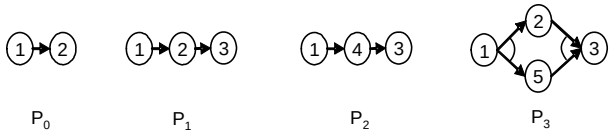


Figure 1. Examples of simple processes

3.1 프로세스 구조 표현

본 연구에서 워크플로우 클러스터링의 분석 대상이 되는 워크플로우 프로세스 저장소는 튜플 $\mathscr{R}$ = <$A^{\diamondsuit}$, W>로 표현할 수 있다. 여기서 $A^{\diamondsuit}$은 워크플로우 프로세스를 수행하는 데 사용될 수 있는 전체 액티비티의 집합이며, W는 워크플로우 프로세스의 집합이다. 본 연구에서는 워크플로우 프로세스의 구조적 유사성을 비교하기 위하여, 워크플로우 프로세스는 특수한 형태의 방향성 그래프로 표현된 튜플 W= <A, T, Split, Join>으로 정의하였다. 여기서 $A \subset A^{\diamondsuit}$는 워크플로우 프로세스에 사용된 액티비티 집합이며, T는 워크플로우 프로세스에서 업무의 선후관계를 나타내는 액티비티 간의 관계 $T \subset (A\text{-}F) \times (A\text{-}\{a_s\})$이다(단, a_s는 단일한 시작 액티비티, $F \subset A$는 종료 액티비티의 집합이다). 그리고 Split와 Join은 각각 분기와 병합을 나타내는 특수한 형태의 통제흐름인 Split : T→{AND, XOR, OR}와 Join : T→{AND, XOR, OR}로 정의할 수 있다.

워크플로우 프로세스 저장소 $\mathscr{R}$에 저장된 워크플로우 W의 총 개수를 N = |W|이라고 하고, N개의 프로세스 내에 포함된 구별되는 액티비티 $A^{\diamondsuit}$의 총 개수를 n = $|A^{\diamondsuit}|$이라고 하자. N개의 프로세스들은 일부 액티비티를 서로 공유하고 있으며, 둘 이상의 액티비티를 동시에 공유하는 경우에는 전이도 공유할 수 있다. 본 연구에서는 워크플로우 프로세스의 구조적 특징을 표현하기 위하여 액티비티 벡터와 전이 벡터를 정의한다.

먼저, 프로세스 P_x ($1 \le x \le$N)의 액티비티 벡터 a_x는 n차 벡터이며, a_x의 i번째 엘리먼트 $a_{i,x}$($1 \le i \le n$)는 i번째 액티비티가 액티비티 프로세스 P_x에서 실행될 확률($e_{i,x}^{act}$)로 표현된다. 프로세스 P_x의 전이 벡터 t_x는 n^2차 벡터이며, t_x의 (i, j)번째 엘리먼트 $t_{ij,x}$($1 \le i, j \le n$)는 전이 실행확률($e_{ij,x}^{tran}$)과 거리 가중치 $1/d_{ij,x}$의 곱이다.

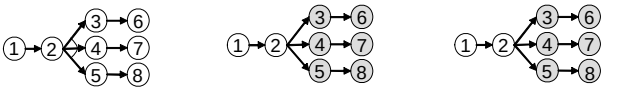
$$a_x = (a_{i,x}),\ a_{i,x} = e_{i,x}^{act},\ where\ i = 1,\cdots,n$$

$$t_x = (t_{ij,x}),\ t_{ij,x} = \frac{1}{d_{ij,x}} e_{ij,x}^{tran},\ where\ i,\ j = 1,\cdots,n$$

아래에서 실행확률과 거리가중치의 정의와 의미에 대하여 자세히 설명한다.

3.1.1 실행확률(execution rate)

먼저 액티비티 벡터와 전이 벡터는 각각 두 프로세스에서 공통으로 발생하는 액티비티 또는 전이를 계산하기 위한 것이다. <Figure 1>의 예에서 프로세스 P_0, P_1, P_2는 직렬 구조를 가지고 있으므로 포함된 모든 액티비티들은 반드시 실행되어야 하며, 그 실행확률이 1이다. 그리고 프로세스 P_3과 P_4와 같이 분기나 병합이 포함된 경우는 Split과 Join의 종류에 따라 의존적이며, 본 연구에서는 AND, XOR, OR로 구분하여 실행확률을 측정한다.



$e_{i,x}^{act} = 1$ $\quad e_{i,x}^{act} = \Pr(a_{i,x}) = \prod_t \frac{1}{s_t}$ $\quad e_{i,x}^{act} = \Pr(a_{i,x}) = \prod_t \frac{1}{2}$

(a) Serial, AND SPlit (b) XOR Split (c) OR Split

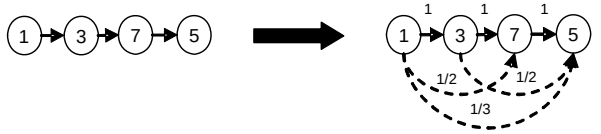
Figure 2. Weighted Complete Dependency Graph (wCDG)

<Figure 2>는 세 가지 종류의 분기 종류(AND, XOR, OR)의 실행확률을 보여준다. AND 분기는 직렬 구조(serial)와 마찬가지로 모든 액티비티와 전이를 진행해야 하므로, 분기 내에 포함된 액티비티의 실행확률을 1로 간주할 수 있다. 반면에 XOR 분기의 경우 분기에 대한 사전지식이 없다면, 각 분기는 동일한 확률로 실행가능성을 가지므로 실행확률은 1/s이다(s는 fan-out의 개수). OR 분기의 경우에 XOR와 마찬가지로 분기에 대한 사전지식이 없다면, 각 분기의 실행가능성은 서로 독립적이고 실행 또는 취소, 두 가지 사건을 가지므로 실행 확률은 1/2로 간주될 수 있다. 만약 XOR나 OR의 경우 프로세스의 과거 실행 결과를 분석하여 실행확률을 예측할 수 있다면, 그 확률값을 액티비티와 전이의 실행확률을 반영할 수 있다. 본 연구에서는 cycle을 갖는 Loop 구조는 가정하지 않는다. Loop에 대한 실행확률의 처리는 사전지식에 의한 실행횟수에 근거하거나 분기의 확률을 고려하되 1.0 이하의 값으로 제한하는 정책 등을 사용할 수 있을 것이다.

3.1.2 거리가중치(distance weight)

<Figure 1>의 예에서 표현된 프로세스의 전이만으로 프로세스 전체의 구조적 유사성을 측정하는 데 충분하지 못하다. 예를 들면, <Figure 1>에서 P_1와 P_2의 액티비티 2와 4와 같은 대체적인 액티비티가 사용되거나, 추가적인 액티비티가 삽입되는 등의 경우에는 직접적인 선후 관계만으로는 전체 프로세스 구조를 반영하기 힘들다. 그래서 본 연구에서는 <Figure 3>과 같이 간접적인 액티비티의 선후관계까지 고려한 완전 의존 그래프(CDG : Complete Dependency Graph)를 이용한다. 직접적

인 전이 관계(explicit transition)에 간접적인 선후 관계를 표현한 전이 관계(implicit transition)를 유추하여 표시한 다음, 유추된 전이에는 거리에 반비례하여 가중치를 준 것이 가중된 완전 의존 그래프(weighted CDG)이다. 이러한 간접적 선후 관계에 대한 고려는 Figure 1의 예에서 P_1와 P_2의 액티비티 1과 3의 예와 같이 직접적인 전이 관계 외에도 간접적인 선후 관계를 고려함으로써 구조적 유사성을 정밀하게 반영할 수 있다.



(a) explicit transitions (b) implicit transitions

Figure 3. Weighted Complete Dependency Graph (wCDG).

3.2 프로세스 유사성 측정

먼저, 워크플로우 프로세스 P의 구조적 유사성 $sim(P_x, P_y)$는 두 가지 척도, 액티비티 유사성 sim_{act}와 전이 유사성 sim_{tran}의 가중평균으로 표현할 수 있다. 조정인자 a는 액티비티 유사성의 반영비율을 나타낸다.

$$sim(P_x, P_y) = \alpha \cdot sim_{act}(P_x, P_y) + (1-\alpha) \cdot sim_{tran}(P_x, P_y)$$

본 연구에서는 프로세스 벡터 간의 유사성을 비교하기 위하여 Cosine 계수(coefficient)를 사용하였다. 두 워크플로우 프로세스의 구조 P_x와 P_y의 유사성을 측정하기 위하여 앞 절에서 정의된 액티비티 벡터 a_x와 a_y와의 Cosine 계수와, 전이 벡터 t_x와 t_y의 Cosine 계수를 사용한다. Cosine 계수는 두 벡터의 동일 엘리먼트 값이 클수록 높은 값을 나타낸다. 다시 말하면, 실행확률이 1인 액티비티(또는 전이)가 많을수록, 실행확률이 낮거나 0인(즉, 실행되지 않은) 액티비티(또는 전이)가 적을수록 액티비티 벡터(또는 전이 벡터) 간의 Cosine 계수는 높은 값을 가진다. 또한, 두 프로세스에서 공통으로 실행되는 액티비티 간의 거리가 가까울수록(거리 가중치가 낮을수록) 전이 벡터 간의 Cosine 계수는 높은 값을 가진다.

두 프로세스 구조 P_x와 P_y의 액티비티 유사성 $sim_{act}(P_x, P_y)$와 전이 유사성 $sim_{tran}(P_x, P_y)$은 아래와 같이 계산된다.

$$sim_{act}(P_x, P_y) = \frac{a_x \cdot a_y}{|a_x||a_y|} = \frac{\sum a_{i,x}\, a_{i,y}}{\sqrt{\sum a_{i,y}^2}\sqrt{\sum a_{i,y}^2}}$$

$$sim_{act}(P_x, P_y) = \frac{t_x \cdot t_y}{|t_x||t_y|} = \frac{\sum t_{ij,x}\, t_{ij,y}}{\sqrt{\sum t_{ij,x}^2}\sqrt{\sum t_{ij,y}^2}}$$

4. 워크플로우 클러스터링

본 연구에서 제안하는 워크플로우 클러스터링은 기업에 누적된 프로세스 저장소를 분석하고 활용하는 도구로 사용될 수 있다. 기업은 워크플로우 클러스터링을 적용하여 기업의 지적 자산인 프로세스를 분석함으로써, 유사한 프로세스 군집을 발견하여 새로운 프로세스 설계시에 참조할 수 있으며, 유사 프로세스 군집 내에 존재하는 프로세스의 특성 및 패턴을 분석하여 기존 프로세스를 개선하는 데 활용할 수 있다. 프로세스 군집의 발견과 유사 프로세스의 추천은 프로세스 마이닝 및 프로세스 패턴 분석, 프로세스 검색 및 질의, 프로세스 최적화 등의 목적으로 다양하게 활용되어 프로세스 설계 및 개선에서 시간과 비용을 절감할 수 있다.

4.1 워크플로우 추천 시스템

본 연구의 워크플로우 클러스터링은 워크플로우 추천 시스템을 위한 모듈로서 고안되었으며, <Figure 4>는 워크플로우 추천 시스템의 구성을 보여주고 있다.

워크플로우 추천 시스템은 크게 세 가지 모듈로 구성된다. 먼저, 프로세스 베이스(Process Base)는 단위 업무의 의미론적 정보와 프로세스의 실행 정보를 저장하고 있다. 이는 도메인 지식을 구조화하고 있는 작업분해구조(Work Break-down Stru-

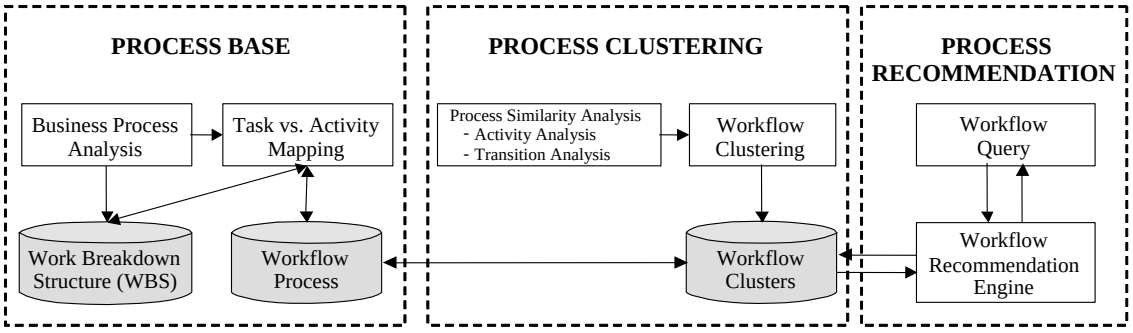


Figure 4. Workflow recommendation system

cture)와, 프로세스 설계를 보관한 프로세스 정의 저장소(Workflow Repository)로 구성된다. 두 번째, 프로세스 클러스터링(Process Clustering) 모듈은 작업분해구조와 프로세스 정의의 두 가지 정보를 바탕으로 프로세스의 의미론적 유사성과 구조적 유사성을 분석하여, 클러스터를 구축하는 역할을 한다. 여기에서 다음 절에 설명될 워크플로우 클러스터링 알고리즘이 사용된다. 마지막으로 프로세스 추천(Process Recommendation) 엔진은 사용자로부터 워크플로우 질의를 받아, 워크플로우 클러스터로부터 적절한 프로세스를 추천하는 역할을 수행한다.

4.2 워크플로우 클러스터링

워크플로우 클러스터링은 다음과 같이 수행한다. 프로세스 도메인 분류에서는 액티비티 유사성에 기반한 클러스터링을 수행한다. 클러스터링 대상이 되는 프로세스 저장소 전체에서 액티비티 및 전이의 종류와 프로세스별 발생 빈도를 고려하여 프로세스 벡터를 생성하고(Frequency Analysis), Cosine 척도에 기반하여 유사성 측정을 수행한 후(Similarity Measurement), 측정된 유사성을 바탕으로 AHC(Agglomerative Hierarchical Clustering) 알고리즘을 사용하여 클러스터링을 수행한다(Process Clustering).

액티비티 벡터와 전이 벡터는 유사성 측정(Similarity Measurement) 단계에서 Cosine 척도에 의하여 프로세스 간의 유사성이 계산된다. 이 유사성 결과값은 프로세스 클러스터링(Process Clustering) 단계에서 AHC 알고리즘에 의하여 군집화된다. AHC 알고리즘과 같은 계층적(hierarchical) 클러스터링 방법은 K-means 알고리즘과 같은 분할적(partitioning) 클러스터링 방법보다 시간적으로는 비효율적이지만, 일반적으로 더 정확한 성능을 보인다(Karypis *et al*., 1999). <Figure 5>은 AHC 알고리즘을 포함한 워크플로우 클러스터링의 전반적인 과정을 보여준다.

먼저, 본 알고리즘은 3.1절에서 제시한 워크플로우 프로세스 저장소 $\mathcal{R} = (A^{\diamondsuit}, W)$을 대상으로 수행된다. $A^{\diamondsuit}$은 액티비티 공간을 의미하며, W는 워크플로우 프로세스 집합으로 튜플 W = <A, T, Split, Join>으로 표현된다. $A \subset A^{\diamondsuit}$는 워크플로우 W의 액티비티 집합, T는 W의 전이 관계, Split와 Join은 분기와 병합을 표현하는 함수이다.

Workflow_clusteirng 알고리즘은 크게 프로세스 벡터 $V^{\mathcal{R}}$를 생성하는 *create_process_vectors* 함수, 프로세스 벡터 간의 유사성 $S(V_f^{\mathcal{R}})$을 측정하는 *vector_similarity* 함수를 포함한다. 클러스터링 알고리즘의 핵심 부분은 각 프로세스를 개별적인 클러스터로 초기화함으로써 시작된다(5행 ~ 6행). 초기화된 N = |W| 개의 군집을 목표한 k개로 군집화 하는 과정을 반복한다(8행~21행). 먼저, 병합을 위해서 *find_nearest_pair(S(C))*를 이용하여 유사성이 가장 큰 두 클러스터 c_u, c_v를 찾은 다음, 클러스터 c_u, c_v를 삭제하고 새로운 병합된 클러스터 c_m를 추가하고(11행), 클러스터 c_u, c_v에 소속되어 있던 프로세스를 새로운 클러스터 c_m으로 소속시킨 후(13행~15행), 새로운 클러스터 c_m와 다른 클러스터 c_i 간의 평균 유사성 $sim(c_m,c_i) = \{|c_u|sim(c_u,c_i)+|c_v|sim(c_v,c_i)\}/(c_u+c_v)$;을 계산하여 유사성 행렬을 업데이트한다(17행~20행). 이러한 병합 과정을 N-k회 실행함으로써 k개의 목표 클러스터를 생성할 수 있다.

Input: A workflow repository $\mathcal{R} = (A^{\diamondsuit}, W)$, the number of clusters *k*, a blending factor α.
Output: A clustering result $C^{\mathcal{R}} = \{(C, M, S)|$ result clusters C, the membership of workflows M(W), similarity matrix between clusters S(C)$\}$.

Algorithm *Workflow_clustering*(**in** ($\mathcal{R}$, *k*, α), **out** C)
1: $V^{\mathcal{R}}$: = *create_process_vectors*($\mathcal{R}$);
2: $S(V^{\mathcal{R}})$: = *vector_similarity*($V^{\mathcal{R}}$);
3: //make initial clusters
4: C : = W;
5: **for** each $w_i \in W$ **do** $M(w_i)=c_i$;
6: S(C) : = $S(V^{\mathcal{R}})$;
7: //agglomerative clustering
8: **while** | C | > *k* **do**
9: (c_u, c_v) = *find_nearest_pair*(S(C));
10: //update clusters
11: *delete*(c_u, c_v, *C*); *add*(c_m, *C*);
12: //update the membership
13: **for** each $w \in W$ **do**
14: **if** $M(w) == c_u \,||\, M(w) == c_v$ **then** $M(w) = c_m$;
15: **end for**
16: //update similarity matrix of the new cluster;
17: **for** each $c_i \in C$ ($i \neq u,v$) **do**
18: $sim(c_m,c_i) = \{|c_u|sim(c_u,c_i) + |c_v|sim(c_v,c_i)\}/ (c_u+c_v)$;
19: $S(C) \leftarrow sim(c_m,c_i)$;
20: **end for**
21: **end while**

Function *create_process_vectors*(**in** $\mathcal{R}$, **out** $V^{\mathcal{R}}$)
for each $w \in W$ **do**
$a_w = \mathbf{0}$; $t_w = \mathbf{0}$;
for each $a^i \in A^{\diamondsuit}$ **do**
if $a^i \in A_w$ **then** $a_{i,w}$=*execution_rate*(*w*, a^i);
end for
for each $a^i, a^j \in A^{\diamondsuit}$ **do**
if $a^i \in A_w$ && $a^j \in A_w$ **then** $t_{ij,w}=a_{i,w}a_{i,w}$distance(*w*, (*i*, *j*));
end for
$A^{\mathcal{R}} \leftarrow a_w$; $T^{\mathcal{R}} \leftarrow t_w$;
end for
return $V^{\mathcal{R}} = (A^{\mathcal{R}}, T^{\mathcal{R}})$;

Function *vector_similarity*(**in** ($V^{\mathcal{R}}$,α), **out** $S(V^{\mathcal{R}})$)
for each $w_x,w_y \in W$ **do**
$sim_{act}(w_x,w_y) = a_x \cdot a_y/|a_x||a_y|$;
$sim_{tran}(w_x,w_y) = t_x \cdot t_y/|t_x||t_y|$;
$sim(w_x,w_y) = \alpha\, sim_{act}(w_x,w_y)+(1-\alpha)sim_{tran}(w_x,w_y)$;
$S(V^{\mathcal{R}}) \leftarrow sim(w_x,w_y)$;
end for
return $S(V^{\mathcal{R}})$;

Figure 5. Hierarchical algorithm for workflow clustering

프로세스 저장소 $\mathcal{R}$의 액티비티 공간 $A^{\diamond}$의 크기를 $n = |A^{\diamond}|$이라고 하고, 워크플로우 집합 W의 크기를 N=|W|이라고 할 때, *Workflow_clusteirng* 알고리즘은 총 $O(n^2N^2)$의 시간 복잡도를 가진다. 클러스터링을 수행하기 이전에 수행되는 *create_process_vector* 함수는 모두 액티비티 벡터에 생성하는 데 $O(nN)$, 전이 벡터를 생성하는 데 $O(n^2N)$의 복잡도를 갖는다. 그리고 프로세스 간의 *vector_similarity*를 계산하는 데, 액티비티 벡터에 대하여 $O(nN^2)$, 전이 벡터에 대하여 $O(n^2N^2)$의 복잡도를 가진다. 이렇게 계산된 프로세스 간의 유사성을 이용하여 agglomerative clustering을 수행하는 과정은 (N-k)회 반복되며, N^2의 비교를 수행하므로 $O(N^3)$의 복잡도를 가진다. 그러므로 워크플로우 클러스터링 알고리즘은 총 $O(n^2N^2) + O(N^3)$의 시간 복잡도를 가진다.

4.3 예제

이 절에서는 워크플로우 클러스터링 과정의 이해를 돕기 위하여 10개의 보험 처리 프로세스를 이용한 예제를 제공한다.

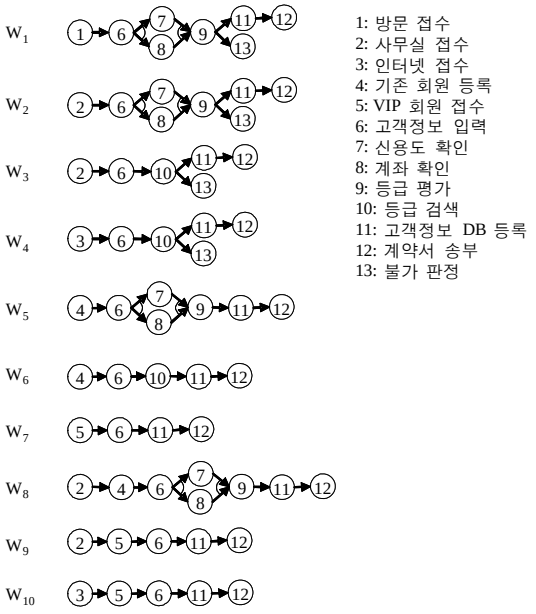


Figure 6. Example of insurance processes

예제는 <Figure 6>과 같이 13가지 액티비티를 사용하는 10개의 보험 처리 프로세스를 사용한다. 보험 처리 프로세스는 W_1에서 W_4와 같이 방문, 사무실, 인터넷으로 접수가 개시될 수 있으며, W_5에서 W_7과 같이 기존 회원이나 VIP 회원을 위한 프로세스도 있다. 그리고 W_8에서 W_{10}은 사무실이나 인터넷에서의 기존 회원과 VIP 회원을 위한 프로세스이다. 이들 프로세스는 접수 창구나 회원의 성격에 따라 신용 등급 평가를 필요로 하거나, 등급 조회만 필요로 하는 경우도 있으며, 일부 검증된 고객에 대해서는 보험 판매 불가 판정을 포함하지 않는다. 이 프로세스 저장소는 액티비티 공간의 크기가 n = 13이고, 워크플로우 집합 W의 크기는 N = 10이다. 그러므로 액티비티 벡터는 13차원이며, 전이 벡터는 169차원이다. 생성된 프로세스 벡터의 예를 보면, W_1의 액티비티 벡터는 a_1 = (1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0.5, 0.5, 0.5)이며, 전이 벡터 t_1 은 $t_{1,6} = t_{6,7} = t_{6,8} = t_{7,9} = t_{8,9} = 1$, $t_{1,7} = t_{1,8} = t_{6,9} = t_{9,11} = t_{8,13} = 1/2$, $t_{1,9} = t_{9,12} = 1/3$, $t_{7,11} = t_{7,12} = t_{8,11} = t_{8,13} = t_{11,12} = 1/4$, $t_{6,11} = t_{6,13} = t_{7,12} = t_{8,12} = 1/6$, $t_{1,11} = t_{1,13} = t_{6,12} = 1/8$, $t_{1,12} = 1/10$으로 non-zero 값의 엘리먼트를 갖는다. 액티비티 벡터와 전이 벡터 생성시, 액티비티 11, 12, 13은 fan-out = 2인 XOR 분기에 포함되어 있으므로 실행 확률이 1/2임을 기억하기 바란다. 이렇게 생성된 13개의 액티비티 벡터와 전이 벡터를 기반으로 조정인자 alpha = 0.5를 사용하여 계산된 프로세스 벡터 간의 유사성은 <Table 2>와 같다. <Table 2>에서 유사성이 가장 높은 프로세스는 W_5와 W_8이므로, 가장 먼저 C_1으로 클러스터링 된다. 계산된 유사성을 바탕으로 k = 1까지 클러스터링을 시행하면, <Figure 7>와 같은 계층적 클러스터링 결과를 얻을 수 있다.

<Figure 7>의 클러스터링 결과를 살펴보면, W_5, W_8, W_1, W_2를 포함하는 C_5는 신용도와 계좌 확인을 한 후 등급 평가를 하는

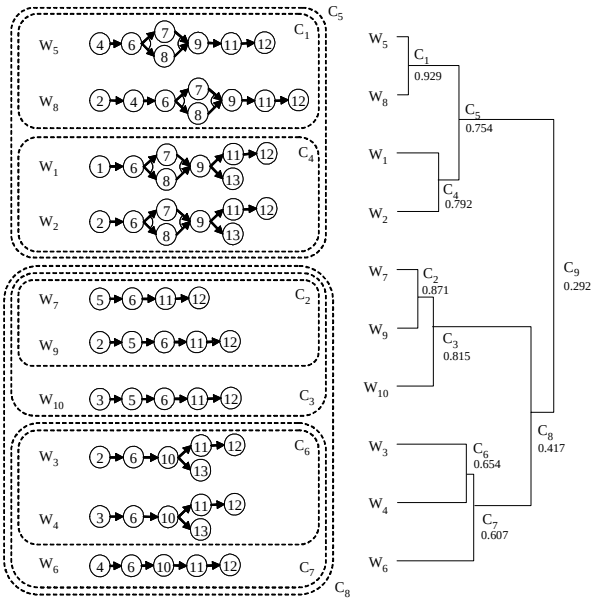


Figure 7. Clustering result of insurance processes

Table 2. Similarities between insurance processes

	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	W_{10}
W_1	1.000	0.792	0.207	0.207	0.748	0.219	0.257	0.695	0.227	0.227
W_2		1.000	0.429	0.207	0.748	0.219	0.257	0.825	0.369	0.227
W_3			1.000	0.654	0.230	0.607	0.345	0.354	0.495	0.305
W_4				1.000	0.230	0.607	0.345	0.215	0.305	0.495
W_5					1.000	0.514	0.411	0.929	0.361	0.361
W_6						1.000	0.531	0.478	0.466	0.466
W_7							1.000	0.382	0.871	0.871
W_8								1.000	0.440	0.336
W_9									1.000	0.759
W_{10}										1.000

프로세스들이며, W_7, W_9, W_{10}을 포함하는 C_3는 VIP 회원 관리를 위한 프로세스들이며, W_3, W_4, W_6을 포함하는 C_7은 고객정보를 입력한 후 등급을 검색하여 결정하는 프로세스들로 군집화 되었음을 알 수 있다.

5. 실험 및 토의

본 연구에서는 워크플로우 클러스터링 모델을 실험하기 위하여 프로세스 구조를 자동 생성하여 실험을 수행하였다. 워크플로우 클러스터링을 실험하는데 있어서, 실제 워크플로우 데이터를 사용하지 않고 모의 데이터를 생성하는 이유는 다음과 같다. 먼저, 워크플로우 클러스터링의 실험을 위하여 다양한 형태의 다수의 프로세스 구조가 필요하지만, 실제로 충분한 양의 워크플로우가 공개되지 않기 때문이다. 뿐만 아니라, 충분한 양의 실제 프로세스를 구한다고 하더라도 의미론을 해석하고 동일성 여부를 판별하는 데 많은 비용과 오차가 발생한다. 마지막으로 가장 중요한 이유는, 실제 프로세스는 실험 계획에서 필요한 파라미터의 제어와 통제가 거의 불가능하기 때문이다. 이러한 이유로 본 연구에서는 통제 가능한 그래프 구조의 실험 프로세스를 자동으로 생성하여 실험에 사용하였다.

본 연구의 실험을 위해 생성한 프로세스는 AND와 XOR를 재귀적으로 포함하고 있는 잘 구조화된 프로세스(well-structured process)를 가정하였다. 구조화된 프로세스란 하나의 시작 액티비티와 하나의 종료 액티비티를 가지며, 분기와 병합이 항상 동시에 일어나는 구조의 프로세스이다(Verbeek *et al.*, 2001). 구조화된 프로세스로 자동 생성된 프로세스는 교착상태와 같은 설계의 무결성 검증을 생략할 수 있으며, 임의로 프로세스 그래프를 생성할 때 발생할 수 있는 불필요한 전이 관계가 포함되지 않기 때문에 모의 프로세스를 자동 생성하기에 용이하다. 본 연구에서는 다음과 같은 과정으로 구조화된 프로세스를 생성하였다. 프로세스 생성에 사용되는 파라미터로는 액티비티 스페이스의 크기(ActSpace), 단위 액티비티의 발생 확률(ProbOfAct), 프로세스의 최대 크기(MaxProcSize)가 사용된다.

먼저, 프로세스는 최대 MaxProcSize 이하의 크기를 갖는 직렬 구조로 생성된다. 직렬 구조를 구성하는 액티비티들은 MaxProcSize의 횟수만큼 ProbOfAct의 확률로서 ActSpace 크기의 액티비티 스페이스에서 액티비티를 임의로 선택한다. 그러므로 생성된 프로세스의 평균 크기는 MaxProcSize×Prob-OfAct 이다. 다음으로 초기 생성된 직렬 구조의 일부를 블록을 구성한다. 하나의 블록을 생성하기 위하여 직렬 구조 중에서 블록에 포함될 액티비티의 개수를 정하여, 임의로 정해진 직렬 구조의 특정 위치에, 임의로 분기의 개수(fan-out size)와 종류(AND 또는 XOR)로 블록을 생성한다. 이러한 블록 생성 과정을 재귀적으로 반복함으로써 직렬 구조는 다단계 블록을 포함한 프로세스 구조로 변경된다.

<Figure 8>은 실험변수 ActSpace = 26, MaxProcSize = 16, ProbOfAct = 0.9 하에서 생성된 모의 프로세스의 예를 보여준다. 이 프로세스는 유사성 분석을 위한 프로세스 인코딩에 의해 qu[ez(r|h|aj)x&wcb]fm로 표현된다. (r|h|aj)는 fan-out의 크기가 3인 OR 블록을 나타내며, [e... x&wcb]은 fan-out의 크기가 2인 AND 블록을 나타낸다.

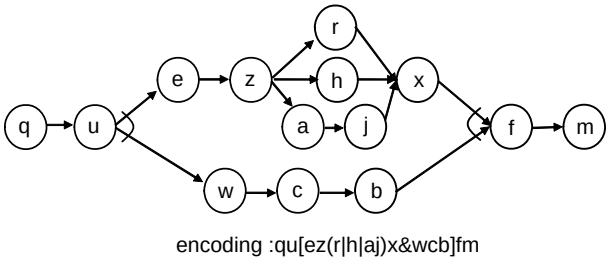


Figure 8. Example of synthetic workflow processes

첫 번째 실험은 샘플 프로세스의 개수와 클러스터의 수에 따른 결과 클러스터들의 유사성을 측정하였다. 유사성은 내부 프로세스들의 평균 유사성(Intra-Similarity)과 외부 클러스터 간의 평균 유사성(Inter-Similarity) 두 가지를 측정하였다. Figure 9는 샘플 프로세스의 개수 N = 50, 100, 200에 대하여 한 클러스터에 포함될 평균 프로세스의 수(N/K)를 3, 5, 10, 20, 50의 다섯 가지로 나누어 클러스터링 한 결과의 내부 유사성(Intra)과 외부 유사성(Inter)을 보여준다. 아래의 실험은 고정변수 ActSpace = 20, MaxProcSize = 7, ProbOfAct = 0.9 하에서 수행한 결과이다.

N/K		3	5	10	20	50
K		17	10	5	3	1
N = 50	Intra	0.368	0.200	0.068	0.065	N/A
	Inter	0.010	0.012	0.016	0.015	N/A
K		33	20	10	5	2
N = 100	Intra	0.343	0.205	0.102	0.070	0.030
	Inter	0.012	0.011	0.011	0.014	0.015
K		67	40	20	10	5
N = 200	Intra	0.338	0.245	0.146	0.085	0.043
	Inter	0.014	0.012	0.013	0.013	0.015

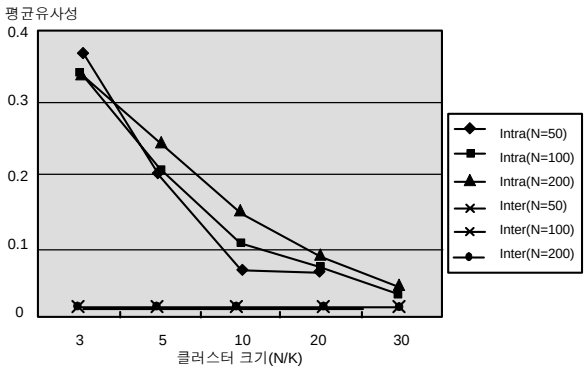


Figure 9. Experimental results of workflow clusters varying N and K.

첫 번째 실험 결과는 샘플 프로세스들의 클러스터 개수(K)가 작아짐에 따라서 외부유사성은 미세하게 증가하는 반면에, 내부유사성은 급속하게 감소하는 것을 보여준다. 이 때, 세 가지 샘플 크기에 대하여, 클러스터 내의 프로세스 평균 개수, 즉 클러스터의 평균 크기가 10일 때 가장 큰 차이를 보이고 있다. 이 실험 조건 하에서 클러스터 크기 10을 기준으로 다음 실험들을 설계하였다.

두 번째 실험은 프로세스의 크기와 액티비티 가능공간의 크기에 따른 프로세스 클러스터 결과의 특성을 알아보기 위한 것이다. 액티비티 가능공간(ActSpace)은 10개, 20개, 50개 세 가지로 설정하였으며, 세 경우에 대하여 각 액티비티 유형이 샘플 프로세스에 나타날 수 있을 확률인 액티비티 탄생 확률(M/A = MaxProcSize/ActSpace)을 1/4, 1/2, 1/0.75, 1/1의 네 가지에 대하여 실험을 수행하였다. 즉, M/A가 클수록 샘플 프로세스들의 평균 크기가 커지며, 서로 공유하는 액티비티도 많도록 설계하였다. <Figure 10>은 두 번째 실험 결과의 내부유사성(Intra)과 외부유사성(Inter)을 보여준다.

두 번째 실험 결과는 프로세스 내의 액티비티 등장 확률(M/A)이 1과 0에 가까워질수록 내부 유사성이 높아지는 경향을 보여준다. 그 이유는 M/A = 1에 가까워지는 경우에는 프로세스가 공유하는 액티비티와 전이가 많아지기 때문이며, 반면 M/A = 0에 가까워지는 경우에는 짧은 크기의 프로세스들로만 구성된 일부 클러스터들의 내부 유사성이 높게 나타나 평균 내부 유사성을 상승시키기 때문으로 해석되었다.

M/A		1/4	1/2	1/0.75	1/1
M		2.5	5	7	10
A=10	Intra	N/A	0.341	0.235	0.263
	Inter	N/A	0.023	0.064	0.112
M		5	10	15	20
A=20	Intra	0.105	0.144	0.342	0.070
	Inter	0.026	0.059	0.102	0.014
M		12	25	37	50
A=50	Intra	0.053	0.084	0.454	0.085
	Inter	0.021	0.045	0.078	0.013

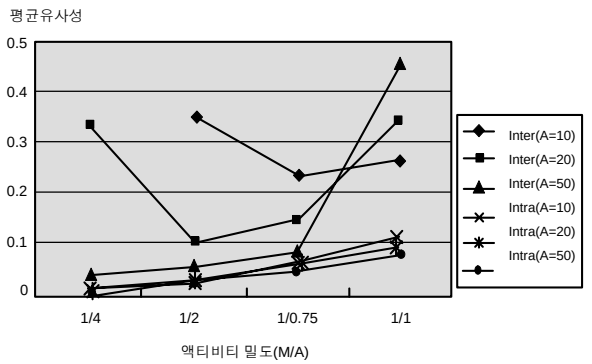


Figure 10. Experimental results of workflow clustering varying MaxProcSize(M) and ActSpace(A)

세 번째 실험은 서로 다른 빈도의 액티비티들로 프로세스를 구성하여 현실의 프로세스 특성을 반영하기 위한 것이다. 현실의 프로세스에는 상대적으로 자주 사용되는 액티비티와 드물게 사용되는 액티비티가 섞여 있다. 이 실험에서는 하나의 프로세스 그룹은 모든 액티비티들이 동일한 탄생 확률(M/A = 0.67)을 갖고, 다른 프로세스 그룹은 액티비티들이 서로 다른 탄생 확률(M/A = 0.33, 0.67, 1.0)을 가지도록 설계하였다. 액티비티 가능공간(ActSpace)의 크기는 20, 프로세스의 최대크기(MaxProcSize)는 10으로 고정하였으며, N/K = 5 하에서 N = 50, 100, 150, 200, 250의 프로세스를 생성하여 실험하였다. <Figure 11>은 세 번째 실험 결과를 보여준다.

<Figure 11>은 액티비티의 탄생 확률을 차별화한 두 번째 그룹(different birth rate)의 평균 클러스터들이 동일한 탄생 확률의 첫 번째 그룹(identical birth rate)보다 내부 유사성이 더 큰 결과를 보여주었다. 그 이유는 두 번째 그룹이 모든 프로세스에 자주 등장하는 액티비티 유형(M/A = 1.0)으로 인해 평균 유사성을 높인 것으로 해석된다. 이 실험을 통하여 실세계의 프로세스는 두 번째 그룹과 같이 액티비티들이 각기 탄생 확률을 가지므로, 모의 프로세스를 사용한 실험 결과보다 더 큰 값의 내부 유사성을 가지며 클러스터링 될 것으로 예상된다.

N		50	100	150	200	250
K		5	10	15	20	25
identical birth rate	Intra	0.188	0.242	0.278	0.302	0.316
	Inter	0.006	0.008	0.010	0.009	0.008
different birth rate	Intra	0.194	0.275	0.312	0.316	0.336
	Inter	0.009	0.010	0.009	0.012	0.010

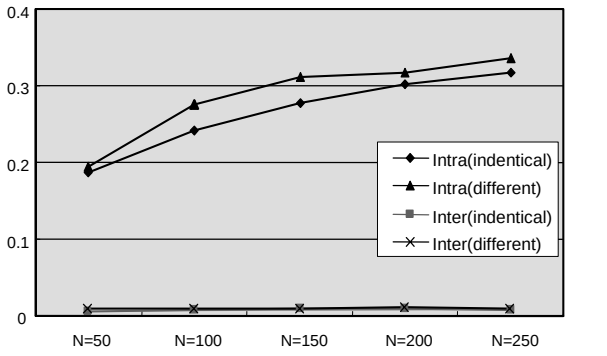


Figure 11. Experimental results of workflow clusters in different activity birth rate

7. 결 론

비즈니스 프로세스는 기업의 핵심 프로세스를 정형화, 표준화한 것으로 기업의 경쟁력 재고를 위한 핵심 자산으로 인식된

다. 앞으로 비즈니스 프로세스는 기업의 자산으로 누적되어 이를 체계적으로 관리하고 개선하는 방법이 필요하다. 본 연구에서는 기업의 비즈니스 프로세스를 체계적으로 관리하기 위하여, 기존의 프로세스 설계들을 구조적 유사성에 기초하여 클러스터링하는 방법론을 제시하였다.

본 연구에서는 워크플로우 클러스터링을 위하여 프로세스 구조 벡터를 정의하여 프로세스 간의 유사성을 비교하는 척도를 제안하였다. 그리고 이 척도를 이용하여 프로세스 간의 유사성을 바탕으로 유사한 구조를 갖는 워크플로우 프로세스들을 군집화하는 알고리즘을 제시하였다. 워크플로우 클러스터링은 4.3절에서 제시된 예제와 같이 기존의 프로세스 설계들을 분류하고 그 구조적 특성과 의미를 유추함으로써, 기존 설계를 개선하거나 신규 설계를 지원하는 데 활용될 수 있다.

본 연구에서 제시된 실험은 워크플로우 클러스터링의 성능과 특징을 파악하기 위해 수행되었으며, 현실에 비하여 액티비티 공간의 크기나 프로세스의 개수가 적다. 하지만, 현실의 프로세스 저장소에 대하여 액티비티 유사성을 이용한 프로세스 클러스터링을 전처리로 수행한다면, 유사한 성격의 프로세스들로 축약되어 분석할 프로세스의 개수와 액티비티 공간의 크기를 급격히 줄일 수 있을 것으로 예상된다. 또한, 이러한 단계적 워크플로우 클러스터링을 적용함으로써, 본 연구에서 제시된 알고리즘의 계산 시간을 상당히 단축할 수 있을 것이다.

프로세스 기반 정보시스템이 확산되면서 프로세스 분석의 중요성이 커지고 있다. 워크플로우 클러스터링과 같은 다양한 프로세스 분석 및 재활용에 관한 연구는 기업의 프로세스 자산을 체계적으로 이해하고 분석하여, 새로운 프로세스를 설계하고 기존 프로세스를 개선하는데 유용하게 사용될 수 있다.

참고문헌

van der Aalst, W. M. P. and Basten, T. (2002), Inheritance of workflows : an approach to tackling problems related to change, *Theoretical Computer Science*, **270**(1), 125-203.

van der Aalst, W. M. P., Hofstede, A. H. M., Kiepuszewski, B., and Barros, A. P. (2003), Workflow Patterns, *Distributed and Parallel Databases*, **14**(1), 5-51.

van der Aalst, W. M. P. and Weijters, A. J. M. M. (2004), Process Mining : a Research Agenda, *Computers in Industry*, **53**(3), 231-244.

Bae, J., Bae, H., Kang, S.-H. and Kim, Y. (2004), Automatic Control of Workflow Processes Using ECA Rules, *IEEE Transactions on Knowledge and Data Engineering*, **16**(8), 1010-1023.

Bae, J.-S., Kwon, B.-C., Jung, J.-Y., and Kang, S.-H. (2005), Workflow Collaboration Design using Similarity Measures among Process Definitions, *Review of Korean Society for Internet Information*, **6**(1), 52-61.

Bae, J., Liu, L., Caverlee, J. and Rouse, W. (2006), Process Mining, Discovery, and Integration using Distance Measures, *Proceedings of International Conference on Web Service 2006*, September 18-22.

Bunke, H. and Shearer, K. (1998), A Graph Distance Metric based on the Maximal Common Subgraph, *Pattern Recognition Letters*, **19**, 255-259.

Bunke, H. (1999), Error correcting graph matching : On the influence of the underlying cost function, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**(9), 917-922.

Bunke, H. and Kandel, A. (2000), Mean and maximum common subgraph of two graphs, *Pattern Recognition Letters*, **21**, 163-168.

Cardoso, J. (2005), How to Measure the Control-flow Complexity of Web processes and Workflows, In: Fischer, L. (ed.) : *Workflow Handbook 2005*, WfMC, Lighthouse Point, FL, 199-212.

Corneil, D. G. and Gotlieb, C. C. (1970), An Efficient Algorithm for Graph Isomorphism, *Journal of the ACM*, **17**(1), 51-64.

Fernandez, M. L. and Valiente, G. (2001), A graph distance metric combining maximum common subgraph and minimum common supergraph, *Pattern Recognition Letters*, **22**, 753-758.

Ha, B., Bae, J., Park, Y. T., and Kang, S-H. (2006), Development of process execution rules for workload balancing on agents, *Data & Knowledge Engineering*, **56**(1), 64-84.

Hammouda, K. M. and Kamel, M. S. (2004), Efficient Phrase-Based Document Indexing for Web Document Clustering, *IEEE Transactions on Knowledge and Data Engineering*, **1**6(10), 1279-1296.

Hur, W., Bae, H., and Kang, S-H. (2003), Customizable Workflow Monitoring, *Concurrent Engineering- Research and Applications*, **11**(4), 313-326.

Jansen-Vullers, M. H., van der Aalst, W. M. P., and Rosemann, M. (2006), Mining Configurable Enterprise Information Systems, *Data and Knowledge Engineering*, **56**(3), 195-244.

Jung, J., Hur, W., Kang, S-H., and Kim, H. (2004), Business Process Choreography for B2B Collaboration, *IEEE Internet Computing*, **8**(1), 37-45.

Karypis, G., Han, E., and Kumar, V. (1999), Chameleon : Hierarchical Clustering Using Dynamic Modeling, *IEEE Computer*, **32**(8), 68-75.

Kim, H., Jung, J.-Y., and Kang, S-H. (2003), Extensible Collaborative Process Composition using Workflow Inheritance, *IE Interfaces*, **16**, 49-54.

Kim, Y., Kang, S-H., Kim, D., Bae, J., and Ju, K.-J. (2000), WW-Flow : Web-Based Workflow Management with Runtime Encapsulation, *IEEE Internet Computing*, **4**(3), 55-64.

Lian, W., Cheung, W. W., Mamoulis, N., and Yiu, S. (2004), An Efficient and Scalable Algorithm for Clustering XML Documents by Structure, *IEEE Transactions on Knowledge and Data Engineering*, **16**(1), 82-96.

Malone, T. W., Crowston, K., and Herman, G. A. (2003), *Organizing Business Knowledge : The MIT Process Handbook*, The MIT Press, Cambridge, MA.

Messmer, B. T. and Bunke, H. (1998), Error-correcting graph isomorphism using decision trees, *Journal of Pattern Recognition and Artificial Intelligence*, **12**, 721-742.

Reijers, H. A. and I. T. P. Vanderfeesten (2004), Cohesion and Coupling Metrics for Workflow Process Design, *Proc. 2nd Int. Conf. on Business Process Management*, 290-305.

RosettaNet (2001), RosettaNet Implementation Framework : Core Specification 2.0. http://www.rosettanet.org/.

Shapiro, R. (2002), A Comparison of XPDL, BPML and BPEL4WS, Cape Visions, White paper.

Schimm, G. (2004), Mining exact models of concurrent workflows, *Computers in Industry*, **53**(3), 265-281.

Simitsis, A., Vassiliadis, P., and Sellis, T. (2005), State-Space Optim-

ization of ETL Workflows, *IEEE Knowledge and Data Engineering*, **17**(10), 1404-1419.

Verbeek, H. M. W., Basten, T., and van der Aalst, W. M. P. (2001), Diagnosing Workflow Processes using Woflan, *The Computer Journal*, **44**(4), 246-279.

WfMC. (1999), *Terminology and Glossary*, WFMC-TC-1011, Workflow Management Coalition, Hampshire, United Kingdom.

WfMC. (2002). *Workflow Process Definition Interface − XML Process Definition Language*, WfMC-TC-1025, Workflow Management Coalition, Hampshire, United Kingdom.

Zamir, O. and Etzioni, O. (1998), Web Document Clustering : A Feasibility Demonstration, *Proc. 21th Int. ACM SIGIR Conf.*, 46-54.

Zhang, K. and Shasha, D. (1989), Simple Fast Algorithms for the Editing Distance between Trees and Related Problems, *SIAM Journal of Computing*, **18**(6), 1245-1262.