최적합 객체 선정을 위한 선 클러스터링 알고리즘*

장주현° 노회영
강원대학교 컴퓨터과학과
jhjang@kangwon.ac.kr°, young@cc.kangwon.ac.kr

Pre-Clustering Algorithm for Selecting Optimal Objects

Joo-Hyun Jang°, Hi-Young Roh
Dept. of CS, Kangwon Nat'l Univ.

요 약

본 논문에서는 절차 중심 소프트웨어를 객체 지향 소프트웨어로 재/역공학기 위한 다단계 절차 중 객체 추출 단계에서 선 클러스터링을 통해 불 필요한 정제 결합단계를 축소하고, 영역 전문가의 선택으로 영역 모델링에 가장 가까운 객체 후보군을 제시하는 알고리즘을 제안하고자 한다. 기존의 연구에서는 영역 모델링과 다중 객체 후보군과의 유사도를 측정하여 영역 전문가에게 최적합 후보를 선택할 수 있는 측정 결과를 제시하였다. 하지만 영역 전문가가 제시하는 영역 모델링이 존재한다면 정제 결합단계이전에 최대한의 선 클러스터링을 통해서 영역 모델링과 가장 유사한 통합 객체를 제시할 수 있고, 정제 결합 단계를 선 클러스터링을 통해서 축소할 수 있으며 이를 통해서 객체 후보군과 영역모델링의 유사도를 향상 시키며 클러스터링에 따른 시간과 공간을 절약할 수 있다. 따라서 본 논문에서는 영역 모델링과 사용자의 함수, 전역변수의 선택을 통해 영역 모델링에 가장 유사한 객체 후보군을 찾는 선 클러스터링 알고리즘 제안 하고자 한다.

1. 서 론

본 논문에서는 최적합 객체 선정을 위한 선 클러스터링 알고리즘을 다룬다. 일반적으로 시스템을 재공학 할 때 유지, 보수 및 재사용적 측면에서 기존의 방식들보다 유리한 객체 지향 파라다임을 적용하여 기존의 시스템으로 재개발한다면, 소프트웨어 생산성을 향상시킬수 있고 소프트웨어의 유지 보수 비용을 절감할 수 있으며, 시스템에 새로운 요구를 수용할 수 있게 되는 등 많은 장점을 가지게 된다.

사용한 객체 추출 방법은 다음과 같다. 1) 전처리 과정에서는 객체 추출을 위한 VPR(Visyal Program Representation)에 기반을 둔 FTV 그래프의 생성/분할을 하고 2) 동적 객체 추출 단계에서는 동적 객체(즉, 클래스의 메소드로 변환될 부분)를 추출한다. 3) 정적 객체 추출 단계에서는 선 클러스터링을 하기 위한 정적 객체(즉, 클래스의 속성으로 변환될 부분)를 추출하며, 2)와 3)의 단계들은 영역 전문가로부터 생성된 영역 모델링이나 1)의 단계를 통해 전 처리된 FTV 그래프에서 함수를 기반으로 하여 클러스터링 될 수 있는 요소들을 제공하고 영역 전문가에게 이를 선택하게 한다. 4) 통합 단계에서는 전 처리 과정이나 선 클러스터링 과정에서 분리된 그래프가 여러 개 존재 한다면 각각의 처리된 그래프를 통합한다. 본 논문에서는 기존의 객체 추출 방법중에 VTF를 기반으로 1개의 객체 후보군이 아닌 가능한 경우의 수에 따른 객체 후보군을 생성하고 이를 정체,결정한후 요구 분석서를 이용한 영역 모델링의 생성과 영역 모델과 클래스 사이의 유사도를 측정하여 사용자에게 다양한 선택 기회를 준 기존 연구방법[1]을 바탕으로 하여 객체 후보군 생성 이후 영역 모델링과의 비교를 통한 객체 후보군 선택이 아닌 영역 모델링이나 사용자의 함수 선택에 따른 선 클러스터링을 한 후에 선 클러스터링 된 객체 후보군을 대상으로 하여 통합,정제를 통해서 사용자에게 좀 더 적합한 객체 후보를 제공하는 알고리즘을 제안하고자 한다.

2. 관련 연구

기존의 논문에서는 객체 추출시 함수의 파라미터, 전역변수, 자료형들간의 관계성을 이용하였고, 이들간의 클러스터링 순서에 따라 다른 결과값이 발생하는 비결정적인 방법을 사용하였고, 1개의 객체 후보군과 1개의 영역 모델링을 비교하였다[2,3,4,5,6,7]. 또한 VTF(Global Variable, Type, Function) 들 간의 관계를 이용하여 객체 추출, 클래스 추출, 클래스들 간의 관계성을 추출한 연구도 있으며, 이 그래프에서 에지의 방향성과 호출/사용에 따른 가중치를 둔 FTV그래프를 이용한 연구도 존재한다[1]. 이 연구에서는 기존의 연구방법[2,3,4,5,6,7]과 함수, 전역변수, 자료형들 간의 관계를 이용한다는 것은 동일하다. 이 연구에서는 1)클러스터링 순서가 고정된 결정론적 방법을 사용, 2)1개의 객체 후보군이 아닌 가능한 경우의 수에 따른 다중 객체 후보군의 생성, 3) 객관적이고 의미가 있는 객체 추출 방법으로의 정제와 결정, 4)요구 분석서를 이용한 영역 모델링의 생성 5) 1개의 객체 후보군에서 다수의 객체 후보와 영역 모델과의 평균 유사도를 제시하여 사용자에게 다양한 기회를 제공하였다. 또한 이 연구에서는 클러스터링 분할을 위한 관계성에 대한 가중치 부여 기준의 객관성, 객관적이고 의미가 있는 객체 추출에 대한 통계적 방법으로의 정제와 결정을 하였으며, 결정단계에서는 역 비순환 그래프를 이용하여 최적 객체군으로부터 사용자가 원하는 정재된 객체군을 선택할 수 있다. 하지만 이 연구에서는 노드들 간의 에지의 중요도를 수치화로 입력하여, 각 노드들간의 최대 관계값 MRV(Maximum Relativity value)를 계산하고 이를 가지고 기본 분할 및 기본결합을 하였다. 아래 그림1은 이 연구에서 사용한 객체 선정 단계를 나타낸 것이다.

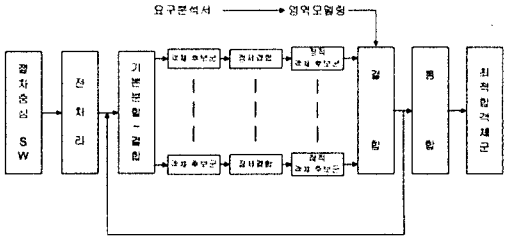

그림 1. 기존의 객체 추출 절차 흐름도

* 본 연구는 첨단정보기술연구센터를 통해 한국과학재단의 지원을 받았음

이는 각 노드들간의 관계값에 따라서 클러스터링 그래프가 변하게 되는 단점을 가지고 있다. 또한 클러스터링이 반복되면서 기본 결합이 불가능한 객체 후보군이 존재 할 수 도 있으며, 이 객체 후보군에 대해서도 MRV연산을 하여 불 필요한 연산에 따른 지연을 발생시킨다.

3. 선 클러스터링 알고리즘

제안하는 선 클러스터링 알고리즘의 수행 과정은 그림 2와 같다. 그림 2의 알고리즘은 기존연구의 알고리즘에서 노드들의 최대 관계값을 가지는 에지를 선택하는 것이 아니라 직접적인 함수를 선택하여 선 클러스터링될 그래프를 구성하고 이 그래프에서 참조되는 전역변수들을 추출하게 된다. 추출된 전역변수들은 정적 객체의 선정 부분에서 사용되게 되며, 그래프를 구성하는 함수 들만이 전역변수를 참조 하면 그 동적 객체군의 정적 객체로 포함되게 된다. 그렇게 않은 경우에는 나중에 정적 객체선정 단계를 통하여 정적 객체 들이 선정되게 된다. 절차 흐름은 다음과 같다. 1) FTV 그래프를 생성하는 FTV그래프 생성단계와 2)클래스의 동적 객체 즉 메소드로 변환될 부분과 어떠한 객체 후보에도 소속이 되지 않는 함수들을 정의하게 된다. 또한 2)의 단계에서 사용되는 변수들중 메소드와 같이 클러스터링 될 변수들을 정의하는 3)의 단계를 거치고 2)와 3)의 단계를 통해 클러스터링 된 동적, 정적 객체들과 클러스터링 되지 않을 동적, 정적 객체들을 분리 한 후에, 선 클러스터링 된 객체 후보들을 가지고 객체 후보간의 통합을 하게 되며, 통합된 객체 후보를 통하여 최적합 객체군을 선정할 수 있다. 다음 그림 2는 객체 추출절차의 흐름도를 나타낸다.

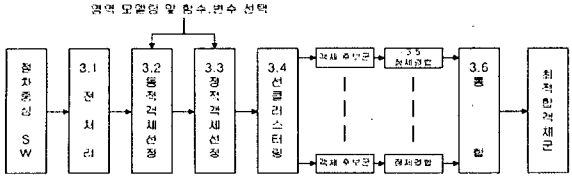

그림2. 객체 추출 절차 흐름도

3.1 전처리

전처리 과정에서는 먼저 가능성이 거의 없는 객체 후보들을 제외한다. 그래프 G^{FTV} 에서는 필요 없이 많은 정보 또는 n개의 노드를 1개의 노드로 처리하여도 문제가 되지 않는 것이 존재 할 수 있다. 이러한 불필요한 정보와 불필요한 노드들을 줄일 수 있다면 객체 추출을 하기 위한 시간과 공간을 절약 할 수 있다. 전 처리 과정은 크게 아래와 같은 4개의 하위 단계로 구성된다. 4개의 하위 단계는 순차적으로 진행하여야 한다. : 1)전역변수, 자료형과 함수로 구성된 그래프 G^{FTV}를 생성하고, 2)그래프가 G^{FTV}가 분리될 수 있을 경우 분리하고, 3)동일한 자료형에 대한 변수들을 1개로 클러스터링하고, 함수와 함수간의 호출관계나 함수의 세 번째 과정의 결과와의 사이에 하나의 에지만이 존재할 경우 1개로 클러스터링한다. 아래 그림3은 Binary-Tree 프로그램의 전처리 결과이다.

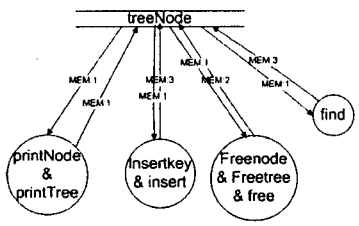

그림3. Binary-Tree 프로그램의 전처리 결과

3.2 동적 객체 선정

동적 객체 선정단계에서는 클러스터링 될 동적 객체를 선정하게 된다. 본 논문에서는 전처리된 동적 객체 후보들을 영역 전문가에게 제시함으로 해서 동적 객체 후보들을 바로 선택하게 하며 또한 어떠한 객체에도 선택되지 않을 함수들, 즉 독립 클래스 될 함수들을 선택함으로 인해서 통합시에 불필요한 연산을 줄일 수 있다. 선정된 동적 객체후보군 DO_K(Dynamic Objects Candidate)과 어떠한 객체에도 선택되지 않을 객체군 NCO(Not Clurstering Objects Candidate)은 다음과 같이 정의 된다.

$$DO_k = \sum_{i=1}^{n} DO_i$$

k= 자연수, n=해당 동적 객체군의 동적 객체수

$$G_i^{NCO} = G^{FTV} - G^{FTV:control} - G^{FTV:isolated\ type} - \sum_{j=1}^{n} G_j^{NCO}$$

i = 해당 객체군, j = i를 제외한 나머지

DO_k 는 하나의 객체로 포함되는 동적 객체군이 되며 영역 전문가가 선택한 동적 객체들의 집합이다. 또한 G_i^{NCO} 는 어느 객체에도 속하지 않고 독립적으로 존재하는 객체 들이며, DO_K와 마찬가지로 사용자가 선택하게 된다.

3.3 정적 객체 추출

정적 객체 추출 단계에서는 선택된 동적 객체에서 호출,기록하는 변수들에 대한 추출 과정을 거치게 된다. 정적 객체에 대한 선택은 동적 객체에서 판독하는 변수들을 대상으로 한다. 이때 DO_k에 속한 함수 들이 같은 변수를 참조하는 경우에는 DO_k 에 포함시키는 정적 객체(Static Object)로 추출하면 된다. 하지만 여러 DO_k들이 하나의 변수들을 참조 하는 경우에는 각각의 DO_k 들과 변수와의 가중치를 계산하여, 정적 객체가 포함될 동적 객체를 선정하게 된다. 변수와의 가중치(Weight of Value)는 0 ~ 1의 값으로 영역전문가에 의해서 결정되게 되며, 함수가 변수를 판독(MEM)하는 횟수에 따라서 결정되게 되며, 최대값을 갖는 DO_k 와 같이 클러스터링하며, 해당 수식은 아래와 같다.

$$SWV^{DOi} = \sum_{i=1}^{n} (DO_i^{mem} \times WV)$$

정적 객체가 클러스터링 되는 동적 객체는 각각의 가중치의 합 SWV^{DOi}들중 최대 MWVS(Maximum SWV)를 갖는 동적 객체군 DO_K 로 통합 된다.

3.4 선 클러스터링

앞의 두 단계를 거치면 동적 객체군이 선정되고, 또한 동적 객체군에서 사용될 정적 객체, 즉 변수들이 선정이 된다. 선정된 동적 객체군과 정적객체군은 선 클러스터링을 통해서 하나의 객체군으로 통합되게 된다. 선 클러스터링은 다음과 같은 세 단계로 구성된다. 1)어느 객체에도 속하지 않는 동적 객체의 분리, 2)선택된 동적 객체군의 클러스터링, 3)동적 객체군에 맞는 정적 객체군의 클러스터링의 순으로 진행된다. 먼저 어느 객체에도 속하지 않는 동적 객체들을 그래프에서 분리하며, 분리된 그래프에서 동적 객체들을 클러스터링한다. 또한 이 과정에서 동적 객체와 정적 객체와의 가중치의 합을 구하여 정적 객체들이 포함될 동적 객체들을 선정하게 되며, 선정된 정적 객체들은 동적 객체 군에 클러스터링 된다. 이에 따른 알고리즘은 아래와 같다.

```
입력 : G^VPR
출력 : G0^CVPR ... Gi^CVPR

G^FTV := createFTV(G^VPR)
G^FTV := removeIsolateType(G^FTV)
G0^TVCC ... Gi^TVCC
:= removeControlFunction(G^FTV)

For each Gi^TVCC
{
clusterSameType(Gj^TVCC)
clusterFunction(Gj^TVCC)
}

select DOk
select NCO

removeNCO(Gj^TVCC)

For each Gi^TVCC
{
DOK : clusterSameDynamicObjects(Gj^TVCC)
calculateSWV(Gj^TVCC, Variable)
clusterStaticObjects(Gj^TVCC)
}
```

그림3. 선 클러스터링

위의 알고리즘1은 전처리 과정을 위해 G^{VPR} 그래프를 생성하며(createFTV), 생성된 그래프에서 독립적 type(removeIsolateType)과 제어 함수(removeControlFunction)를 먼저 제거 한다. 제거된 G_i^{TVCC} 그래프에서 같은 type(clusterSameType)과 통합될 수 있는 함수(clusterFunction)들을 클러스터 한후 어느 객체에도 속하지 않는 객체 후보군을 먼저 제거(removeNCO)한 후 제거된 그래프에서 동적 객체를 추출(clusterSameDynamicObjects)하게 된다. 동적 객체가 추출된 후에 정적 객체들을 추출(clusterStaticObjects)하기 위하여 동적 객체와의 가중치 합(calculateSWV)을 구한 뒤 얻어진 가중치 합을 이용하여 정적 객체를 추출하게 된다. 그림3의 전처리된 Binary-Tree 프로그램의 경우에 printNode와 printTree의 노드, Freenode & Freetree & free의노드, insertkey 와 insert를 하나의 동적 객체 후보로 선택한 후에 모든 가중치를 0.9로 동일하게 선택하면 아래 그림과 같이 동적, 객체와 정적 객체가 선택되는 선 클러스터링이 이루어 진다.

그림 4. Binary-Tree 프로그램의 선 클러스터링 결과

4. 결론 및 향후 연구과제

본 논문에서는 최적합 객체 추출을 하기 위해 영역 전문가의 영역 모델링과 함수 선택을 이용하여, 동적 객체와 정적 객체를 선택하여 영역 전문가가 원하는 객체를 추출하기 위한 선 클러스터링 알고리즘을 제안하였다. 기존의 논문에서는 객체 추출시 함수의 파라미터, 전역변수, 자료형들간의 관계성을 이용, 이들간의 클러스터링 순서에 따라 다른 결과 값이 발생하는 비결정적인 방법을 사용한 논문들[2,3,4,5,6,7]이 있었으며, 이를 결정론적 방법을 사용하여 1개의 객체 후보군이 아닌 다중의 객체 후보군을 생성하고 이를 요구 분석서를 이용하여 영역 모델링과의 유사도 측정 방법을 사용한 연구[1]가 있었다. 본 논문에서는 [1]에서 연구한 방법에서 영역 모델과의 유사성을 높이는 객체 추출방법을 위해 영역 전문가의 요구 분석서나 함수의 선택을 통한 방법으로 선 클러스터링을 하는 알고리즘을 제안하였다. 또한 선 클러스터링을 함으로써 정제 결합단계를 축소함에 따라 객체 추출을 위한 시간과 공간을 절약하는 알고리즘을 제안한다. 본 논문의 향후 연구과제로는 실질적인 알고리즘의 구현과 구현된 알고리즘을 예제 프로그램에 적용하여, 도출된 결과가 기존의 연구에 비해 높은 유사도를 내는 것을 증명해야 하며, 또한 객체 추출을 통해 얻어진 객체 후보군을 이용하여 클래스 추출, 상속관계 추출등 이후의 재공학 절차에 대한 연구를 진행해야 할 것이다.

5. 참고문헌

[1] 박성옥, 노경주, 이문근 "최적합 객체 선정을 위한 다중 객체군 추출", 한국 정보 과학회 논문집(B), 제 26권, 12호, pp.1468 - 1481, 1999년 12월

[2] C. L. Ong and W. T. Tsai, "Class and object extraction from imperative," Journal of Object-Oriented Programming, pp. 58-68, March-April, 1993

[3] E. Horowitz, "A Expensive view of reusable software," IEEE Transaction on Software Engineering, Vol. SE-10 No. 5, Sept. 1984

[4] Harald C. Gall, Rene R. Klosch and Roland T. Mittermier, "Object-Oriented Re-Architecturing," Proc. European Software Engineering Conference, Sept. 1995

[5] Harald C. Gall, Rene R. Klosch and Roland T. Mittermier, "Architecture Transformation of Legacy System," Technical Report Number CS95-418, Seattle, April, 1995

[6] G.Canfora, A. Cimitile and M. Munro, "An Improved Algorithm for Identifying Object in Code," Software-Practive and Experience, Vol. 26(1), pp. 25-48 Jan. 1996 T.

[7] Panos E. Livadas and Theodore Johnson, "A New Approach to Finding Objects in Programs," Journal of Software Maintenance : Research and Practice, Vol. 6, pp. 249-260, 1994