추상 필터를 이용한 논리 프로그램의 상향식 수행 개선 : 실험적 고찰

(Improving Bottom-up Execution of Logic Program using Abstract Filters : An Experimental Approach)

김 문 정 † 창 병 모 †† 최 광 무 †††

(Moon-Jung Kim) (Byeong-Mo Chang) (Kwang-Moo Choe)

요 약 논리 프로그램의 상향식 수행은 논리적 완전성(completeness) 및 비논리적 특성 등의 장점이 있으나 주어진 질의(query)를 답하는데 불필요한 유도를 함으로써 비효율적일 수 있다. 이러한 문제점을 해결하기 위하여 제안된 필터링 방법은 상향식 수행에 기초하여 필터를 이용하여 불필요한 유도 과정을 제한한다. 추상 필터는 이러한 문제를 해결하기 위해 제안된 새로운 필터로써 컴파일 시간에 두 페이즈 추상해석에 의해서 계산된다. 본 논문에서는 추상 필터를 구현하고 실험을 통해서 추상 필터가 기존의 정적 필터보다 강력함을 보였다.

Abstract Bottom-up evaluation of logic programs has a number of advantages such as logical completeness and freedom from non-logical features. However, bottom-up query evaluation may be inefficient because many irrelevant facts may be generated for a given query. The filtering strategy has been designed to overcome this drawback by restricting possible irrelevant facts from being processed in the bottom-up evaluation. A new filter called abstract filter is presented, which is computed by a two-phase abstract interpretation at compile time. In this paper, by experiments, the abstract filter is shown to be more powerful than the static filter.

1. 서 론

서술 논리에 기반을 두고 있는 논리 프로그래밍은 소개된 이후로 하나의 프로그래밍 패러다임으로서 널리 사용되어져 왔다. 특히 Prolog와 같이 논리 프로그램에 관한 실제적인 시스템이 개발되면서 논리 프로그램의 효율적인 구현 및 수행은 주요한 연구 분야를 이루게 되었다. 이에 따라 프로그램에 대한 분석을 통해 얻어낸 정보를 통하여 프로그램의 수행 효율을 꾀하는 여러 가지 분석 기법들이 제안되었는데 Cousot은 여러가지 프로그램 분석 기법들간의 근본적인 유사성을 인식하고, 통합된 하나의 틀로서 추상 해석(abstract interpretation)이라는 수학적 모델을 제시한 바 있다[6]. Cousot가 제시한 추상 해석은 더 구체적인 응용 분야에 적용될 수 있도록 여러 분석 기법으로 구체화되었으며 최근에는 특히 컴파일러 최적화에서의 유용성으로 인해 주목받고 있다[4,7,11].

논리 프로그램에 있어서 Cousot의 추상 해석 기법를 기반으로 많은 기본 틀들과 응용들이 제시되어 왔다[2,3,7,14,15]. 이 중 Barbuti, Giacobazzi, Levi가 제시한 분석 기법은 논리 프로그램의 표준 시맨틱스에 대한 근사값을 구하는 상향 추상 해석이다[2]. 이 상향 추상 해석을 확장하여 더 나은 분석 효과를 얻기위한 두 페이즈 추상 해석(two-phase abstract interpretation)이 저자에 의해서 제시되었다[4,5]. [2]의 상향 추상 해석이 시맨틱 객체로서 애톰을 다루고 있는데 비해 두 페이즈 추상 해석은 애톰뿐 아니라 절까지도 시맨틱 객체로 다루고 있으며 주어진 질의에 대한 관련성을 고려함으로

· 본 연구는 한국과학재단 특정기초연구(과제번호95-0100-54-3) 및 한국전자통신연구원 학연공동연구(과제번호 97-322)의 지원에 의한 것임

† 비 회 원 : (주)퓨처시스템 정보통신연구소
†† 종신회원 : 숙명여자대학교 전산학과 교수
††† 종신회원 : 한국과학기술원 전산학과 교수

논문접수 : 1995년 8월 21일
심사완료 : 1997년 7월 30일

Copyright (C) 2005 NuriMedia Co., Ltd.

써 분석의 응용 효과를 높이고자 하였다. 두 페이즈 추상 해석은 상향 페이즈(bottom-up phase)와 하향 페이즈(top-down phase)로 구성된다. 두 페이즈 추상 해석의 분석 결과는 논리 프로그램의 수행 성능을 향상시키는데 사용될 수 있는데 본 논문에서는 논리 프로그램의 상향식 수행에 적용하여 질의에 관련없는 많은 사실들의 생성을 제한한다. 상향식 수행의 이러한 문제점을 극복하기 위한 여러가지 기법[1,12,16] 들이 제시되었는데 본 논문에서는 필터링 방법을 기반으로 하여 두 페이즈 추상 해석의 결과를 필터링 방법에 적용한다. 이러한 방법을 추상 필터링이라고 한다. 추상 필터링에서 사용되는 추상 필터는 Kifer와 Lozinskii가 제안한 정적 필터[16,17]보다 필터링 효과면에서 강력함이 이론적으로 증명되었다[5,6].

한편 이러한 두 페이즈 추상 해석의 응용에 있어서 두 페이즈 추상 해석의 분석 결과가 실제 논리 프로그램의 수행에 있어 어느 정도의 향상을 가져오는지 평가할 필요가 있다. 상향식 수행에서 추상 필터링을 통한 최적화 효과는 이에 대한 오버헤드와 함께 평가되어야 한다. 본 논문에서는 추상 필터를 이용한 논리 프로그램의 상향식 수행기를 구현하여 필터를 사용하지 않는 세미나이브(seminaive) 수행 방법, 정적 필터링 방법등과 실험을 통해서 비교, 분석한다.

본 논문의 구성은 다음과 같다. 2절에서는 논리 프로그램 및 추상 해석에 관한 기본 개념을 설명하고 3절에서는 두 페이즈 추상 해석에 대한 연구 결과를 간략히 기술한다(이 연구 결과에 대한 이론적 증명은 [5,6]을 참조하기 바람). 4절에서는 추상 필터 계산 과정을 기술하고 5절에서는 구현 및 실험 결과를 기술한다. 6절에서 결론을 맺는다.

2. 관련 연구

본 절에서는 본 논문의 내용을 전개해 나가는데 필요한 논리 프로그램에 관한 기본 개념 및 용어를기술하고 추상 해석에 대해 간단히 살펴본다.

논리 프로그램을 기술하는 언어는 서술 논리에 기반을 두고 있다. 본 논문에서는 다음과 같이 정의되는 일차 언어(first order language)를 전제로 이론을 기술한다[13]. 일차 언어는 (Π, Σ, Var)로 표현되는데, 여기서 Π는 프레디킷(predicate) 심볼들의 집합, Σ는 함수 심볼들의 집합, Var은 변수들의 집합을 나타낸다. 각각의 프레디킷 심볼 $p \in \Pi$와 함수 심볼 $f \in \Sigma$의 인자의 수는 $p/n, f/n$으로 나타낸다. Σ의 함수 심볼과 Var의 변수로부터 구성될 수 있는 모든 텀(term)들의 집합은 $Term(\Sigma, Var)$ 또는 간단히 $Term$으로 나타낸다. $p/n \in \Pi$와 $t_1 \cdots t_n \in Term$으로부터 구성될 수 있는 모든 애톰(atom)들의 집합을 $Atoms$로 나타낸다. $Atoms$의 애톰들로 구성되는 절들의 집합을 $Clause(\Pi, \Sigma, Var)$ 또는 간단히 $Clause$로 나타낸다. 문법 객체(텀, 애톰, 절 등) t에 나타나는 모든 변수들의 집합은 $var(t)$로써 나타낸다. $pred(\delta, i)$는 규칙 δ의 i번째 프레디킷를 나타내고 $head(\delta)$는 규칙 δ의 머리 프레디킷을 나타낸다. n_δ는 규칙 δ의 본체 애톰의 수를 나타낸다. 치환(substitution) θ는 Var에서 $Term$으로의 함수로 정의된다. 이는 임의의 문법 객체로 확장하여 적용될 수 있다. 치환 θ의 객체 t에 대한 적용은 $t\theta$로 표기한다. 부분 함수(partial function) mgu는 한 쌍의 문법 객체를 그 객체들에 대한 최범통합자(most general unifier)로 사상하는 함수로 정의되며 $\theta = mgu(s, t)$는 s와 t가 단일화(unification)될 수 있음을 의미한다. mgu는 일반적으로 등식의 집합들에 대해 확장되어 적용되는데 $\{a_1 = b_1, \ldots, a_n = b_n\}$에 대한 최범통합자는 $mgu(\langle a_1, \ldots, a_n \rangle, \langle b_1, \ldots, b_n \rangle)$라 표기한다. $mgu(\langle\rangle, \langle\rangle) = \epsilon$이다. 임의의 집합 S와 S상에서의 동치 관계(equivalence relation) $\sim$에 대해서, $S/\sim$는 $\sim$에 대한 S상에서의 동치 클래스들의 집합을 나타낸다. 한 요소 $a \in S$에 대해 $[a]_\sim$는 $\sim$에 대한 a의 동치 클래스를 나타낸다.

프로그램에 대한 추상 해석(*abstract interpretation*)[1]이란 추상 도메인 상에서 그 프로그램의 표준 시맨틱스에 대한 근사값을 구하는 것을 말한다. 추상 해석의 기본 개념은 실제 무한할 수도 있는 구체적 도메인(concrete domain)대신 더 단순하고, 일반적으로 유한한 추상 도메인 상에서 프로그램을 수행시킴으로써, 프로그램의 실제 수행시 동작에 대한 정적 정보를 알아내는 것이다. 이러한 정보는 보통 컴파일러에서 오류 발견, 코드 최적화 등에 이용된다[4,8,11]. 추상 해석은 보통 추상 도메인에 대한 고정점(fixed point) 계산으로 이루어진다. 따라서, 추상 도메인은 유한한 래티스(lattice)이거나, 무한 래티스인 경우에는 무한 상승 체인(infinite ascending chain)을 허용하지 않는 래티스인 것이 보통이다. 추상 해석에서는 알아내고자 하는 실제 수행시의 동작을 나타낼 수 있는 구체적 시맨틱스(concrete semantics)를 정의하여 이를 기반으로 수행시 동작에 대한 근사값을 구하게 된다. 이 표준 틀은 구체적 시맨틱스가 최소 고정점의 특성을 갖는 것을 전제

Copyright (C) 2005 NuriMedia Co., Ltd.

로 한다. 전체적인 추상 해석의 방법은 다음과 같이 요약될 수 있다.

(1) 알아내고자 하는 특성을 올바로 나타낼 수 있도록 구체적 시맨틱스를 정한다.

(2) 시맨틱 객체와 연산들의 근사된 기술(approximated description)을 통해 추상 시맨틱스를 정의한다.

먼저, 프로그램 P에 대한 구체적 시맨틱스(concrete semantics)는 구체적 도메인 E 상에서의 단조 연산자(monotone operator) $E_P: E \to E$를 이용하여 정의될 수 있다고 가정한다.

정의 1 : 추상 해석(abstract interpretation)

추상 해석 $((E, \sqsubseteq), E_P, (D, \preceq), D_P, \alpha, \gamma)$은 아래 조건을 만족하는, 완전 래티스(complete lattice) $(E, \sqsubseteq)$, 단조 연산자 $E_P: E \to E$, 완전 래티스 $(D, \preceq)$, 단조 연산자 $D_P: D \to D$, 추상화 함수(abstraction function) $\alpha: E \to D$, 그리고 구체화 함수(concretization function) $\gamma: D \to E$로 구성된다 [6].

(1) α와 γ는 단조 함수이다.

(2) 모든 $d \in D$에 대해 $d = \alpha(\gamma(d))$.

(3) 모든 $e \in E$에 대해 $e \sqsubseteq \gamma(\alpha(e))$이다.

(4) 모든 $d \in D$에 대해 $E_b(\gamma(d)) \sqsubseteq \gamma(D_b(d))$

위의 조건들 중 (1) - (3)이 만족되면 $((E, \sqsubseteq), \alpha, (D, \preceq), \gamma)$은 *Galois insertion*이라 한다. 조건 (4)는 D_P가 E_P를 충실히 수용함을 보장하기 위한 안전성(*safeness*)의 기준이 된다. 여기서, Galois insertions은 구체적 시맨틱 객체들과 추상 시맨틱 객체들간의 대응관계를 나타내며, 또한 고정점 계산을 통한 근사값 계산에서의 정확성을 뒷받침하기 위한 이론적 근거를 마련한다(조건 (4)참조).

추상 해석은 시맨틱스를 기반으로 한다. 따라서 어떻게 시맨틱스를 정의하느냐에 따라 여러 다른 프로그램의 분석 방법을 얻게 된다[3,4,7,15]. 논리 프로그램의 상향식 분석 방법은 주로 최소 고정점 시맨틱스를 기반으로 한다[2]. 상향 추상 해석에 기반이 되는 구체적 시맨틱스는 정의 2의 연산자 T_F의 최소 고정점으로 결정된다. 확장된 Herbrand 유니버스(universe) U_F는 $Term(\Sigma, Var)/_{\sim}$으로 정의되며 $\sim$은 변이(*variance*) 릴레이션 (즉, $t_1 \sim t_2$ iff $\exists \upsilon_{1,\upsilon_2} \mid t_1\upsilon_1 = t_2 \wedge t_2\upsilon_2 = t_1$)이다. 변이 릴레이션은 임의의 문법 객체로 쉽게 확장될 수 있다. 확장된 Herbrand 베이스(base) B_F는 $Atom/_{\sim}$로 정의되며 $\sim$은 애톰에 관한 변이 릴레이션이다. 인터프리테이션(*Interpretation*) I는 B_F의 한 부분 집합으로 정의된다. 임의의 애톰에 대해서 $[a]_{\sim}$와 변수들의 유한 집합 V가 주어졌을 때, V의 변수를 포함하지 않는 대표 원소(representative) a'는 항상 $[a]_{\sim}$에 존재한다. 문법 객체 s와 인터프리테이션 I에 대해, $a_1, \ldots, a_n$ 각각이 s나 $a_i (0 \le i \le n)$들 상호간에 공통 변수가 없는 대표 원소들일 때, $\langle a_1, \ldots, a_n \rangle \ll_s I$ 이라고 표기한다.

정의 2 : T_F

임의의 프로그램 P 에 대해 $T_P: \wp(B_P) \to \wp(B_P)$는 다음과 같이 정의되는 변환(transformation) 연산자이다 [9].

$$T_P(I) = \left\{ h\vartheta \;\middle|\; \begin{array}{l} c = h \leftarrow a_1, \ldots, a_n \in P \\ \langle a'_1, \ldots, a'_n \rangle \ll_c I \\ \vartheta = mgu(\langle a_1, \ldots, a_n \rangle, \langle a'_1, \ldots, a'_n \rangle) \end{array} \right\}$$

프로그램 P 의 시맨틱스는 T_F의 최소 고정점 즉, $lfp(T_P) = T_P^{\omega}(\phi)$으로 결정된다[7]. 논리 프로그램의 상향식 수행은 T_F의 최소 고정점을 구하는 과정으로 프로그램의 사실들로부터 규칙을 이용하여 유도될 수 있는 모든 가능한 사실들을 유도해냄으로써 주어진 질의에 대한 해를 구하는 방법이다. 필터링은 상향식 수행에 필터를 이용하여 데이타 흐름을 제어함으로써 불필요한 사실들의 생성을 제한하는 방법으로써 필터는 일반적으로 텀들의 튜플에 대해 참 또는 거짓을 결정하는 논리식이다. 수행 중 발생되는 튜플에 대해 적용하고자 하는 필터가 그 튜플에 대해 참일 때만 통과시킴으로써 데이타 흐름을 제어하게 된다. 필터 함수 F는 모든 규칙의 각 본체 애톰에 대해 필터를 사상하는 함수로서, 규칙 δ의 i번째 본체 애톰에 대한 필터는 $F(\delta, i)$ 또는 $F_{\delta,i}$로 표기한다. 애톰 a가 필터 $F_{\delta,i}$를 만족할 경우, $a \in F_{\delta,i}$로 표기한다. 필터를 이용한 상향 수행은 아래와 같이 정의된 T_P^F의 최소 고정점, $lfp(T_P^F)$를 계산한다.

정의 3 프로그램 P와 필터 함수 F에 대하여, $T_P^F: \wp(B_P) \to \wp(B_P)$는 다음과 같이 정의된다.

$$T_P^F(I) = \left\{ h\vartheta \;\middle|\; \begin{array}{l} c = h \leftarrow a_1, \ldots, a_n \in P \\ \langle b_1, \ldots, b_n \rangle \ll_c I, b_i \in F_{c,i} (1 \le i \le n) \\ \vartheta = mgu(\langle a_1, \ldots, a_n \rangle, \langle a'_1, \ldots, a'_n \rangle) \end{array} \right\}$$

T_P^F는 프로그램의 표준 시맨틱스를 정의하는데 사용

Copyright (C) 2005 NuriMedia Co., Ltd.

되는 T_F를 수정하여 정의된 것으로 결국 $lfp(T_P^F)$는 $lfp(T_P)$의 부분 집합을 이루게 된다. 프로그램 F에 대해 질의 Q가 주어졌을 때, $lfp(T_P^F)$로부터의 해가 $lfp(T_P)$로부터의 해와 같으면 $lfp(T_P^F)$는 $lfp(T_P)$와 Q에 대해 동등하다고 하며, 이 경우 $lfp(T_P^F)$는 F와 Q에 대해 완전하다고 한다[4,5].

본 논문에서는 새로 제안된 추상 필터를 이용하며 이를 일반적으로 정의하면 다음과 같다.

정의 4 F는 프로그램을 δ는 규칙을 i는 규칙 δ에 대한 인덱스라고 가정하자. 추상 필터 $AF_{\delta,i}$는 F에 대한 추상 인터프리테이션으로 정의되는데 $\gamma(I^A)$는 프레디킷 $pred(\delta,i)$를 갖는 애톰으로만 구성된다.

3. 두 페이즈 추상 해석

이 절에서는 두 페이즈 추상 해석 틀[4,5]을 간단히 설명한다. 두 페이즈 추상 해석은 상향 페이즈(bottom-up phase)와 그 다음에 적용되는 하향 페이즈(top-down phase)로 구성된다. 상향 페이즈는 [2]에서의 기본 상향 추상 해석을 확장하여 시맨틱 객체로서 추상 절을 고려하도록 설계되었다. 이는 추상 절을 고려함으로써 실제 수행 모델들에 적용될 때 더 높은 분석 효과를 꾀할 수 있기 때문이다. 확장된 상향 추상 해석은 모든 가능한 질의에 대하여 프로그램 절들의 성공 패턴들에 대한 근사값을 구한다. 따라서, 질의와 독립적인(query independent) 성격을 갖는다. 하향 페이즈는 상향 페이즈의 결과로부터, 절들의 추상 성공 패턴들 중 주어진 질의에 관련된(*relevant*) 패턴들을 선택한다. 따라서, 두 페이즈 추상 해석은 절들의 성공 패턴들에 대한 근사값 중 질의에 관련된 패턴들을 구하게 된다.

두 페이즈 추상 해석의 상향 페이즈는 [2]의 기본 틀을 확장하여 시맨틱 객체로 절을 고려한다. 집합 C를 $Clause/_\sim$즉 절들의 변이 릴레이션 $\sim$에 대한 동치 클래스들의 집합이라 하자. 이제 인터프리테이션은 $\wp(C)$의 원소 즉 절들의 집합으로 정의한다. 시맨틱 객체로 절을 포함하도록 T_F를 확장한 구체적 연산자 U_F의 정의는 다음과 같다.

정의 5 F 가 논리 프로그램일 때 $U_F: \wp(C)\to\wp(C)$는 다음과 같이 정의된다.

$$U_P(I)=\left\{[c\sigma]_\sim \;\middle|\; \begin{array}{l} c=h\leftarrow a_1,\ldots,a_n\in P \\ \langle h_1\leftarrow b_1,\ldots,h_n\leftarrow b_n\rangle\langle\langle_c I, \\ \sigma=mgu(\langle a_1,\ldots,a_n\rangle,\langle h_1,\ldots,h_n\rangle) \end{array}\right\}$$

프로그램 P의 구체적 시맨틱스는 $lfp(U_P)=U_F^\omega$ 이다.

상향 페이즈는 추상 해석 $((\wp(C),\subseteq),U_P^A(AInt,\sqsubseteq),U_P^A,\alpha,\gamma)$ 으로 정의된다. 여기서는 추상 인터프리테이션의 도메인 $AInt$을 정의한다. 이 도메인은 각 추상 인터프리테이션이 절들의 집합을 나타내고 $(Int,\alpha,AInt,\gamma)$가 Galois insertion이 되는 추상 인터프리테이션의 완전 래티스로 정의된다. $LC=Clause(\Pi,\emptyset,Var)/_\sim$는 flat 절(절내에 함수와 상수가 없고 변수가 반복되서 나타나지 않는 절)의 집합이라고 하자. 추상 치환의 도메인 $(ASub,\sqsubseteq)$은 $(\wp(Sub),\alpha_S,ASub,\gamma_S)$가 Galois insertion을 이루는 완전 래티스이다. 추상 인터프리테이션의 추상 도메인은 LC의 각 대표절에 추상 치환을 조합함으로서 구성한다. 추상 절(abstract clause)은 $LC\times ASub$ 내의 하나의 순서쌍이며 추상 도메인은 $AInt=\wp(LC\times ASub)/_\equiv$으로 $\equiv$는 $\wp(LC\times ASub)$에 대한 릴레이션 $\leq: I_1^A\leq I_2^A$ *iff* $\gamma(I_1^A)\subseteq\gamma(I_2^A)$에 의해서 유도된 동치 릴레이션이다.

정의 6 $(\wp(Sub),\alpha_S,ASub,\gamma_S)$는 추상 치환의 도메인이라고 하자. 이로부터 유도된 추상 인터프리테이션의 도메인 $(Int,\alpha,AInt,\gamma)$는 다음과 같이 구성된다.

1) 함수 $\gamma:\wp(LC\times ASub)\to Int$는 $\gamma(I^A)=\{[c\theta]_\sim\mid\langle c,x\rangle\in I^A,\theta\in\gamma_S(x)\}$로 정의된다.
2) 함수 는 $\alpha(I)=\{\langle c,x\rangle\mid c\in LC, x=\alpha_S(\{mgu(c,c')\mid c'\langle\langle_c I\})\}$로 정의된다.
3) $AInt=\wp(LC\times ASub)/_\equiv$으로 $\equiv$는 $\wp(LC\times ASub)$에 대한 릴레이션 $\leq: I_1^A\leq I_2^A$ *iff* $\gamma(I_1^A)\subseteq\gamma(I_2^A)$에 의해 유도된 동치 릴레이션이며 상응하는 $AInt$에 대한 부분 순서 릴레이션은 $\sqsubseteq$로 표시한다.
4) 함수 γ는 $\gamma([I^A]_\equiv)=\gamma(I^A)$로 함으로써 $\gamma: AInt\to Int$로 확장되고 함수 α는 $\alpha(I)=[\alpha(I)]_\equiv$로 함으로써 $\alpha: Int\to AInt$으로 확장된다.

$(\wp(Sub),\alpha_S,ASub,\gamma_S)$가 Galois insertion이면 추상 인터프리테이션의 유도된 도메인 $(Int,\alpha,AInt,\gamma)$도 Galois insertion이다. 추상 연산자 정의을 위하여 먼저 *추상 단일화 함수* mgu^A를 다음과 같이 정의한다.

정의 7: *추상 단일화 함수*(abstract unification function)

$mgu^A:(Atom^*\times ASub)\times(Atom\times ASub)^*\to ASub\bigcup\{fail\}$

는 다음 조건을 만족하면 정확하다고(correct) 한다.

(1) $k=mgu^A(\langle\langle a_1,\ldots,a_n\rangle;k_0\rangle,\langle\langle h_1;k_1\rangle,\ldots,$

Copyright (C) 2005 NuriMedia Co., Ltd.

$\langle h_n; k_n \rangle \rangle), \theta_i \in \gamma_S(k_i), (0 \le i \le n)$ 이고

(2) $\theta = mgu(\langle a_1, \ldots, a_n \rangle \theta_0, \langle h_1\theta_1, \ldots, h_n\theta_n \rangle)$ 이면 $\theta \in \gamma_s(k)$ 이다 [7].

앞으로 추상 단일화 함수 mgu^A는 정확하고 단조증가한다고 가정한다.

정의 8 U_P^A : $AInt \to AInt$는 다음과 같이 정의된다.

$$U_P^A(I^A) = \bigsqcup \left\{ \langle c, k \rangle \;\middle|\; \begin{array}{l} \langle c = h \leftarrow a_1, \ldots, a_n; k_0 \rangle \\ \langle\langle h_1 \leftarrow \overline{b_1}; k_1 \rangle, \ldots, \langle h_n \leftarrow \overline{b_n}; k_n \rangle\rangle \ll_c I^A \\ k = mgu^A(\langle\langle a_1, \ldots, a_n \rangle; k_0 \rangle, \langle\langle h_1; k_1 \rangle, \ldots, \langle h_n, k_n \rangle\rangle) \end{array} \right\}$$

상향 페이즈는 추상 시맨틱스 $lfp(U_P^A)$를 계산한다. 이 계산은 유한 시간안에 종료되고 $lfp(U_P^A)$는 각 절의 성공 패턴에 대한 근사값을 나타내는 모든 추상 인스터스(instance)들로써 프로그램에 대한 추상 모델을 제공한다 즉 임의의 프로그램 P에 대하여, $\gamma(lfp(U_P^A)) \supseteq lfp(U_P)$이 성립한다[4]

여기서 위의 두 페이즈 추상 해석의 예를 [3]의 깊이 k 추상화에 기초해서 제시한다. 함수 $\rho(t)$는 텀 t를 깊이 k 추상 텀(*depth k abstract term*)으로 사상하는 함수로써 텀 t의 깊이 k 추상 텀은 t에 나타나는 모든 레벨 k 서브텀들을 새로운 변수(*fresh variable*)로 치환함으로써 얻어지는 텀이다. 추상 치환 도메인은 $(Sub, \alpha_\phi, ASub, \gamma_\phi)$으로 정의된다. 주어진 치환 $\vartheta = \{t_1/x_1, \ldots, t_n/x_n\}$에 대해서 $\alpha_\phi(\vartheta)$는 $\alpha_\phi(\vartheta) = \vartheta^k = \{\rho(t_1)/x_1, \ldots, \rho(t_n)/x_n\}$로 정의된다. α_ϕ에 대응되는 구체화 함수는 $(\alpha_\phi, \gamma_\phi)$가 Galois insertion를 이루는 γ_ϕ가 된다. 추상 해석의 도메인은 정의6에 의해서 유도될 수 있으며 이 경우에 추상 단일화 함수는 $mgu^A(\langle a_1, \ldots, a_n; x_0 \rangle, \langle\langle h_1, x_1 \rangle, \ldots, \langle h_n, x_n \rangle\rangle) = \alpha_\phi(\vartheta)$ 이며 $\vartheta = mgu(\langle (a_1, \ldots, a_n) x_0 \rangle, \langle h_1 x_1, \ldots, h_n x_n \rangle)$이다. mgu^A가 정확하고 단조 증가함은 쉽게 증명할 수 있다.

예제 1 다음 프로그램 P에 대해

δ_1: accept(Q,[X|Xs]) ← trans(Q,X,Q1),
accept(Q1,Xs).

δ_2: accept(A,[]) ← final(Q).

trans(q0,a,q1). trans(q0,c,q2). trans(q1,b,q1).
trnas(q1,c,q2). trans(q2,d,q3). final(q3).,

깊이 2인 추상화 함수를 이용했을 때 $lfp(U_P^A)$의 고정점 계산 결과는 다음과 같다.

$lfp(U_P^A)$ = {trans(q0,a,q1).,trans(q0,c,q2).,trans(q1,b,q1).,
trnas(q1,c,q2) ,trans(q2,d,q3).,final(q3).,
accept(q3,[ ]) ← final(q3).,
accept(q2,[d]) ← trans(q2,d,q3),accept(q3,[ ]).,
accept(q0,[a|_]) ← trans(q0,a,q1),accept(q1,[c|_]).,
accept(q0,[a|_]) ← trans(q0,a,q1),accept(q1,[b|_]).,
accept(q0,[c|_]) ← trans(q0,c,q2),accept(q2,[d]),
accept(q1,[c|_]) ← trnas(q1,c,q2),accept(q2,[d]).,
accept(q1,[b|_]) ← trans(q1,b,q1),accept(q1,[c|_]).,
accept(q1,[b|_]) ← trans(q1,b,q1),accept(q1,[b|_]). }

상향 추상 페이즈는 질의와는 상관없이 근사 성공 패턴들(approximated success patterns)을 계산한다. 따라서, 계산된 패턴들 중 많은 것들이 주어진 질의와 관련이 없게 된다.상향 페이즈 다음에 이루어지는 하향 페이즈는 상향 페이즈의 결과에 대해 $lfp(U_P^A)$의 부분 집합을 찾아내게 되는데, 이 집합은 프로그램 절들의 추상 성공 패턴들 중 질의와 관련된 패턴들로 이루어진다([4,5] 참조). 하향 페이즈에 대한 추상 해석은 $((p(C), \subseteq), D_{P_\phi}, (AInt, \sqsubseteq), D_{P_\phi}^A, \alpha, \gamma)$으로 정의된다. 구체적 연산자 D_{P_ϕ}는 하향 페이즈에 대한 구체적 시맨틱스를 위해 다음과 같이 정의된다.

정의 9 D_{P_ϕ} : $p(C) \to p(C)$는 다음과 같이 정의된다. 주어진 질의 Q에 대하여,

$$D_{P_\phi}(I) = \left\{ [c]_\sim \;\middle|\; \begin{array}{l} [c]_\sim = [h \leftarrow \overline{b}]_\sim \in lfp(U_P), \\ mgu(h, a) \neq fail \text{ for some } [a]_\sim \in body(Q \cup I) \end{array} \right\}$$

이며 $body(I)$는 I에 속하는 절들의 본체 애톰들의 집합이다.

하향 페이즈에 대한 구체적 시맨틱스는 $lfp(D_{P_\phi})$로 결정된다. 하향 페이즈에서의 추상 연산자 $lfp(D_{P_\phi}^A)$는 다음과 같이 정의된다.

정의 9 $D_{P_\phi}^A$: $AInt \to AInt$는 다음과 같이 정의된다. 주어진 질의 Q에 대하여,

$$D_{P_\phi}^A(I^A) = \bigsqcup \left\{ \langle c^A \rangle \;\middle|\; \begin{array}{l} c^A = \langle h \leftarrow \overline{b}, k \rangle \\ mgu^A(\langle h; k \rangle, \langle a; \hat{k} \rangle) \neq fail \text{ for some } \langle a; \hat{k} \rangle \in body(a(Q) \cup I^A) \end{array} \right\}$$

이며 $body(I^A)$은 정의 9에서와 같다.

Copyright (C) 2005 NuriMedia Co., Ltd.

$lfp(D^A_{P_Q})$는 $lfp(U^A_{P_Q})$에 속하는 추상 절들 가운데 질의에 관련된 모든 추상 절들을 포함하게 된다. 최소 고정점 $lfp(D^A_{P_Q})$ 계산은 유한 시간에 종료되고 하향 페이즈에 대한 정확하다 즉 임의의 프로그램 P와 질의 Q에 대하여, $lfp(D_{P_Q}) \subseteq \gamma(lfp(D^A_{P_Q}))$ [4]

예제 2 예제 1의 프로그램에 대하여 질의 *accept(q0, [c|X])*가 주어졌다고 할 때 예제 2에 대한 하향 페이즈 분석 결과는 다음과 같다.

$lfp(D^A_P)$= { accept(q3,[ ]) ← final(q3),
accept(q2,[d]) ← trans(q2,d,q3),accept(q3,[ ]).,
accept(q0,[c|_]) ← trans(q0,c,q2),accept(q2,[d]).,
final(q3)., trans(q2,d,q3)., trans(q0,c,q2). }

4. 추상 필터링

두 페이즈 추상 해석의 결과를 필터로 사용하는 필터링 기법을 추상 필터링이라 한다. 그리고 이 때의 필터를 추상 필터라 하는데, 추상 필터는 수행 중 분석으로부터 얻은 근사값에 속하지 않는 사실들의 흐름을 제한한다. 이러한 사실들은 성공 패턴에 속하지 않거나, 성공 패턴에 속한다고 하더라도 질의에 관련된 것이 아니기 때문이다. 추상 필터는 두 페이즈 추상 해석의 결과인 $lfp(D^A_{P_Q})$로부터 정의된다. $lfp(D^A_{P_Q})$의 추상 패턴 중 규칙 δ에 대한 추상 패턴의 집합을 $lfp(D^A_{P_Q})[\delta]$로 표기하고, $lfp(D^A_{P_Q})[\delta]$에 속하는 i번째 본체 애톰들의 집합은 $lfp(D^A_{P_Q})[\delta][i]$로 표기한다.

정의 10 P_Q는 프로그램과 질의 Q를 나타낸다. 규칙 δ와 인덱스 i에 대하여, 추상 필터 는 $lfp(D^A_{P_Q})[\delta][i]$로 정의된다. 만약 애톰 A가 $\gamma(lfp(D^A_{P_Q})[\delta][i])$에 속하면 A는 추상 필터 $AF_{\delta,i}$를 만족한다고 하며, $A \in AF_{\delta,i}$로 표기한다.

예제 3 예제 1의 프로그램에 대한 추상 필터는 예제 2의 결과로부터 다음과 같이 계산된다.

$AF_{\delta_{1,1}}$ = {trans(q2,d,q3),trans(q0,c,q2)}

$AF_{\delta_{1,2}}$ = {accept(q3,[ ]),accept(q2,[d])}

$AF_{\delta_{2,1}}$ = {final(q3)}

T^{AF}_P 를 정의 3에서와 같이 추상 필터 AF에 대해 정의된 연산자라 할 때, 정리 1은 추상 필터를 이용한 상향식 수행의 완전성을 증명한다. 이는 추상 필터를 이용한 상향식 수행에서 구해진 해가 세미나이브 수행의 해와 동일함을 의미한다.

정리 1 프로그램 P에 대한 질의 Q와 추상 필터 AF에 대해, $lfp(T^{AF}_P)$는 질의 Q에 대해 $lfp(T_P)$와 동등하다.

5. 구현 및 실험 결과 분석

필터를 이용한 상향식 수행은 Kifer와 Lozinskii에 의해 필터를 적용시킨 시스템 그래프로써 모델링되었다 [16,17]. 시스템 그래프는 프로그램 상의 규칙과 프레디킷들 간의 데이타 흐름을 표현하는 그래프로서, 논리 프로그램의 상향식 수행을 자연스럽게 모델링하여 주어 상향 수행 모델로서 많이 사용되어져왔다. 시스템 그래프의 정의는 다음과 같다.

정의 9 : 시스템 그래프 SG_F

프로그램 F에 대한 시스템 그래프 SG_F는 $(V_p, V_r, E_{p,r}, E_{r,p}, F)$으로 정의되며 각각은 다음과 같다.

V_p : F의 프레디킷들의 집합

V_r : F의 규칙들의 집합

$E_{p,r} = \{(p,\delta) / i | p \in V_p, \delta \in V_r, p = pred(\delta, i), 1 \le i \le n_\delta\}$

$E_{r,p} = \{(\delta, p) | \delta \in V_r, p \in V_p, p = head(\delta)\}$

F : 모든 아크 $(p,\delta)/i \in E_{p,r}$에 대한 필터를 사상하는 필터 함수

시스템 그래프의 노드는 프로그램의 프레디킷과 규칙들에 대응되는데, V_p의 노드를 *pred-node*, V_r의 노드를 *rule-node*라 한다. 또한 시스템 그래프에서 rule-node는 타원으로 pred-node는 사각형으로 구분하여 나타낸다. $(p,\delta)/i \in E_{p,r}$는 pred-node p로부터 rule-node δ로의 i번째 입력 포트(입력 아크)를 나타낸다. 포트 $(p,\delta)/i$에 대한 필터 함수 F의 값은 그 포트에 대한 필터를 나타낸다. $E_{p,r}$의 각 포트에 대한 필터는 필터를 만족하는 데이타만 통과시킴으로써 그 포트를 통과하는 데이타를 제한하게 된다. 프로그램 F에 대해 필터 함수 F를 갖는 시스템 그래프는 SG^F_P로 나타낸다. 예제 1의 프로그램에 대한 시스템 그래프 SG^F_P는 그림 1과 같다.

시스템 그래프 상에서의 상향식 수행은 그래프 상의 아크를 따른 데이타 흐름을 이용함으로써 질의에 대한 해를 구하게 된다. 시스템 그래프 상에서의 세미나이브

Copyright (C) 2005 NuriMedia Co., Ltd.

상향 수행은 논리 프로그램의 기본적인 상향 수행으로서 $lfp(T_P)$를 계산하게 되며, 다음과 같은 알고리즘으로 나타낼 수 있다. 시스템 그래프의 pred-node 중 입력 포트가 없는 것을 기초 pred-node라 하며, 그렇지 않은 경우를 유도pred-node라 한다. 각 기초 pred-node는 처음에 프로그램의 사실들 중 자신의 프레디킷에 대한 사실들을 가지고 있는 것으로 가정한다.

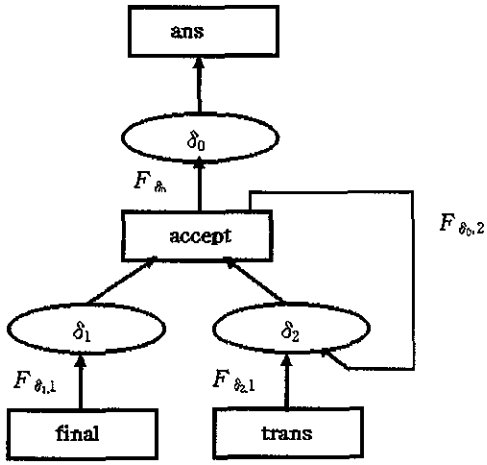


그림 1 예제 1의 프로그램에 대한 시스템 그래프 SG_P^F

알고리즘 1 : SG_F상에서의 세미나이브 수행

1. 각 기초 pred-node는 자신의 튜플들을 출력 포트로 보낸다.
2. SG_F의 노드에 변화가 없을 때까지 2.1과 2.2를 반복한다.
   2.1. 각 rule-node는 입력 포트로부터 새로운 튜플을 버퍼에 저장한 후, 단일화를 수행하여 새로 생성된 사실들를 출력 포트로 보낸다.
   2.2. 각 유도 pred-node는 입력 포트로부터 새로운 튜플을 저장하고 다시 출력 포트로 보낸다.

위의 알고리즘은 필터를 이용하여 $E_{\delta,f}$를 통과하는 데이타 흐름을 제한하도록 쉽게 변경될 수 있다즉, 각 pred-node에서 튜플을 출력 포트로 보낼 때 해당 포트의 필터를 만족하지 않는 튜플은 통과시키지 않게 한다.

이 연구에서는 논리 프로그램의 상향식 수행을 위한 시스템을 구현하였으며 전체적인 구조는 그림 2와 같다. 이를 위해서 논리 프로그램에 대한 문법을 정의하고 이에 대한 파서를 lex와 yacc을 이용하여 구현하였으며 필터를 계산하기 위하여 정적 필터 계산기[12]와 추상 필터 계산기를 구현하였다. 그리고 논리 프로그램의 수행을 위한 세미나이브 상향 수행기와 필터를 이용한 상향 수행기를 구현하였다. 구현은 Concurrent C를 사용하여 SUN3/60 워크스테이션 상에서 이루어졌다. 상향식 수행기는 병렬 수행기이며 시스템 그래프 상의 각 노드는 Concurrent C의 프로세스로서 구현되었고, 노드들 간의 통신은 메시지 전달로써 이루어진다.

본 실험은 두 페이즈 추상 해석에서 추상화 함수로서 깊이 k추상화를 사용하고 있다. 따라서 두 페이즈 추상 해석의 효과를 올바로 평가하기 위하여, 입력 프로그램으로서 함수와 리스트를 사용하는 논리 프로그램들을 사용한다. 이는 함수나 리스트를 사용하지 않는 프로그램에 대한 두 페이즈 추상 해석의 결과는 그 자체가 질의에 대한 해가 되기 때문이다. 이와 같은 전제하에 논리 프로그램의 유형을 살펴본다. 본 실험에서는 다음과 같은 프로그램들을 입력 프로그램으로 사용하였다. 각 프로그램의 기능은 다음과 같다.

(1) path1, path2 : 그래프 상에서 출발점에서 목적지까지의 가능한 경로를 모두 구하는 프로그램으로 각각 질의의 형태만 다르다. path1의 질의가 path2의 질의보다 구체적인 텀의 형태를 갖는다.

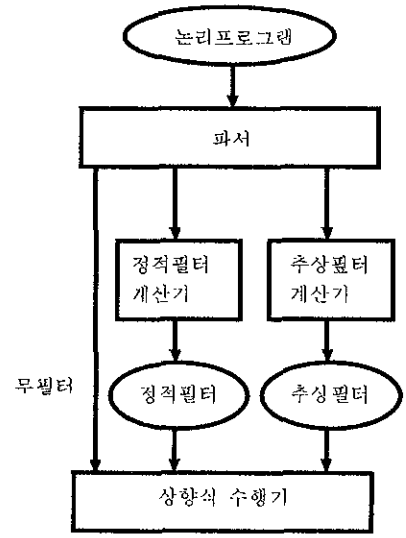


그림 2 시스템 전체 구성도

(2) nfa1, nfa2 : 특정 문자열을 인식하는 비결정적 유한 오토마타로서 각각 질의의 형태만 다르다. nfa1의 질의가 nfa2의 질의보다 구체적인 텀의 형태를 갖는다.

Copyright (C) 2005 NuriMedia Co., Ltd.

(3) map coloring1, map coloring2 : 주어진 map에서 인접된 면에 다른 색을 갖도록 하는 color 패턴을 구하는 프로그램으로서 각각 질의의 형태만 다르다. map coloring1의 질의가 map coloring2의 질의보다 구체적인 텀의 형태를 갖는다.

(4) ackermann : ackermann 함수의 값을 구하는 프로그램

(5) *hanoi* : 하노이 탑을 옮기는 방법을 찾는 프로그램

다음은 실험 분석을 위해 살펴본 프로그램의 유형을 기준으로 두 페이즈 추상 해석 분석 결과의 적용 효과를 비교 분석한다. 상향식 수행에서는 평가 기준으로서 시스템 그래프의 모든 노드에서 발생되는 총 튜플 수를 사용하였다. 추상 필터링은 수행시 발생되는 튜플의 수를 줄이고자하는 의도로 제안되었고, 또 논리 프로그램의 상향식 수행의 성능은 수행시 생성되는 튜플 수로써 좌우된다고 할 수 있으므로 이러한 평가 기준은 타당하다. 상향식 수행은 세가지 수행 방법에 대해 실험되었다. 하나는 필터를 사용하지 않는 세미나이브 수행 방법이고 다른 하나는 정적 필터를 사용하는 정적 필터링[12]이고 나머지 하나는 추상 필터를 사용하는 추상 필터링이다. 또 추상화 깊이에 따른 효과를 알아보기 위해 추상화 깊이가 2,3,4일 때에 대하여 실험하였다 실험에 대한 결과는 표 1과 같으며 괄호 안의 숫자는 세미나이브 수행에 대한 비율이다.

질의의 형태 면에서 위의 결과를 비교해볼 때, path2, nfa2, map coloring2보다는 각각에 대해 질의의 형태가 더 구체적인 path1, nfa1, map coloring1이 정적 필터링과 추상 필터링의 효과가 모두 큰 것으로 나타났다. 또한 추상 필터링의 효과는 정적 필터링보다도 크게 나타났다. 논리 프로그램의 상향식 수행은 그 프로그램에 대해 유도될 수 있는 모든 사실들을 생성해내게 된다. 필터는 이런 모든 가능한 사실들 중에서도 질의에 관련된 사실들만을 생성해내고자하는 목적으로 적용된다. 따라서 보다 구체적인 형태의 질의일수록 가능한 사실들 중 질의에 관련된 사실는 보다 적은 부분에 해당될 것이다. 필터링의 효과가 전혀 없었던 map coloring2의 경우는 프로그램에서 가능한 모든 사실들이 질의에 대한 해를 구하는데 필요했기 때문에 필터에 의해 제한된 사실들이 없었던 것이다. 따라서 필터링의 효과는 보다 구체적인 질의의 형태를 가진 프로그램에 대해 더 크다고 할 수 있다. 사실들의 비중에 따른 프로그램 유형에 따라 필터링의 효과를 비교해볼 때, 사실의 비중이 적은 ackermann과 hanoi는 추상 필터링의 효과가 정적 필터링과 비슷하게 나타났다. 정적 필터링에 사용되는 정적 필터는 프로그램의 사실들의 정보는 고려하지 않고 질의와 rule에 나타나는 정보를 이용하여 계산된다. 반면 추상 필터는 정적 필터의 이러한 한계에 대한 대안으로 제시된 것으로 사실의 정보를 추상 필터의 계산에 고려하고 있다. 따라서 사실의 비중이 작은 프로그램에 있어서는 정적 필터링과 추상 필터링의 효과의 차이는 크지 않을 것으로 예상할 수 있다. 전체적으로 추상 필터링은 정적 필터링 이상으로 튜플 수를 줄임을 알 수 있다. 이에 관하여는 이론적으로 깊이 k 추상 도메인 상에서 추상 필터가 적어도 정적 필터만큼 강력하다는 것이 증명되어 있다 [4]. 또한 추상화 깊이가 클수록 정적 필터링과 추상 필터일의 효과가 더 큰 것으로 나타났다.

표 1 상향식 수행에서의 총 튜플 수

프로그램	seminaive 수행	추상화 깊이	정적 필터링	추상 필터링
path1	376 (1.00)	2	285 (0.76)	197 (0.52)
		3	285 (0.76)	197 (0.52)
		4	285 (0.76)	197 (0.52)
path2	467 (1.00)	2	467 (1.00)	432 (0.93)
		3	467 (1.00)	432 (0.93)
		4	467 (1.00)	432 (0.93)
nfa1	385 (1.00)	2	291 (0.81)	252 (0.70)
		3	271 (0.76)	150 (0.42)
		4	269 (0.75)	148 (0.41)
nfa2	447 (1.00)	2	447 (1.00)	410 (0.92)
		3	447 (1.00)	410 (0.92)
		4	447 (1.00)	410 (0.92)
map coloring1	178 (1.00)	2	164 (0.92)	164 (0.92)
		3	123 (0.69)	102 (0.57)
		4	123 (0.69)	84 (0.47)
hanoi	69 (1.00)	2	63 (0.91)	63 (0.91)
		3	60 (0.87)	60 (0.87)
		4	57 (0.83)	57 (0.83)
parser	1027 (1.00)	2	301 (0.29)	104 (0.10)
		3	91 (0.09)	39 (0.04)
		4	91 (0.09)	26 (0.03)

6. 결 론

본 논문에서는 두 페이즈 추상 해석을 이용하여 논리 프로그램의 상향식 수행기를 구현하고,실험을 통하여 두 페이즈 추상 해석 결과를 수행에 적용함으로써 얻을 수 있는 효과를 분석하였다. 논리 프로그램의 상향식 수행기는 시스템 그래프를 이용한 병렬 수행기로서, 기본적인 상향식 수행인 seminaive 수행과 정적 필터링 방법과 두 페이즈 추상 해석을 이용한 추상 필터링 방법에 대해 수행 중 발생하는 총 튜플 수와 수행 시간을 비교

Copyright (C) 2005 NuriMedia Co., Ltd.

함으로써 두 페이즈 추상 해석이 수행에 미치는 효과를 분석했다. 추상 필터링은 질의에 나타나는 term과 단일화할 수 있는 ground term의 수가 적을수록 수행 중 충 튜플 수를 크게 감소시키고 이에 따라 수행 시간 역시 단축됨을 알았다. 또한 프로그램의 사실 비중이 작은 경우는 정적 필터링의 효과와 비슷하였다. 일반적으로 필터링의 효과는 추상화 깊이가 클수록 더 큰 것으로 나타났다.

앞으로의 연구에서는 두 페이즈 추상 해석기의 분석 결과를 이용하여 Prolog같은 논리 프로그램의 하향식 수행기의 수행 효율을 최적화하는 연구를 수행할 것이다. 두 페이즈 추상 해석은 논리 프로그램의 하향 수행 모델에 분석 결과를 적용하여 수행 중 탐색 공간을 줄임으로써 더 향상된 모델을 가져올 수 있다. 특히 선형 수행 모델과 AND/OR 프로세스 모델에 두 페이즈 추상 해석을 응용한 모델을 제안할 것이다.

참 고 문 헌

[1] Bancilhon, F. and Ramakrishnan, R., An amateur's introduction to recursive query processing strategies, Proc. SIGMOD Int. Conf. on Management of Data, 1986, pp. 16--52.

[2] Barbuti, R., Giacobazzi R., and Levi, G. A general framework for semantics-based bottom-up abstract interpretation of logic programs, *ACM Transactions on Programming Languages and Systems*, 15:133--181 (1993).

[3] Bruynooghe, M. and Janssens, G., An instance of abstract interpretation integrating type and mode inference, *Proc. of 5th Int. Conf. and Symp. on Logic Programmings*, MIT Press, 1988, 669--683

[4] Chang, B.-M., Efficient Bottom-up Execution of Logic Programs using Compile-time Analysis, Ph.D dissertation, Department of Computer Science, KAIST, Feb., 1994

[5] Chang, B.-M., Choe, K.-M and Giacobazzi, R., Abstract Filters : Improving bottom-up execution of logic programs by two-phase abstract interpretation, *Proceedings of the 1994 ACM Symposium on Applied Computing*, ACM Press, March, 1994, pp. 388--393

[6] Cousot, P., and Cousot., R. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixed points, *Proc. of Fourth ACM Symp. Principles of Programming Languages*, 1977, pp. 238-252.

[7] Debray, S.K. and Warren, D.S., Automatic mode inference for Prolog programs, *Journal of Logic Programming Vol. 10*, 1988, pp. 207--229

[8] Demoen, B., Vanroy, P. and Willems, Y.D., Improving the execution speed of compiled Prolog with modes, clause selection and determinism, *Proc. TAPSOFT 1987*, Vol. 250, Springer-Verlag, Berlin, 1987, pp. 111--125 Lecture Notes in Computer Science

[9] Falaschi, M., Levi, G., Palamidessi, C. and Martelli, M., Declarative modeling of the operational behavior of logic languages, *Theoretical Computer Science*, 69: 289-318, 1989

[10] Gallagher, J., Codish, M. and Shapiro, E., Specialization of Prolog and FCP programs using abstract interpretation, *New Gener. Comput., Vol. 6*, 1988, pp. 159--186

[11] Kanamori, T and Kawamura, T, Abstract interpretation based on OLDT resolution, *Journal of Logic Programming, Vol. 15*, 1993, pp. 1--30

[12] Kifer, M. and Lozinskii, E.L., SYGRAF: Implementing logic programs in a database style, *IEEE Trans. on Software Engineering*, 1988, pp. 922--935

[13] Lloyd, J.W., *Foundations of logic programming*, Springer-Verlag, 1984

[14] Marriott, K., and Sondergaard, H. Bottom-up abstract interpretation of logic programs. In *Proceedings of the 5th International Conference on Logic Programming*, The MIT Press, Cambridge, Mass., 1988, pp. 733--748.

[15] Marriott, K. and Sondergaard, H., Semantic-based dataflow analysis of logic programs, *Information Processing 89*, 1989, pp. 601--606

[16] Ramakrishnan, R., Magic Templates: A spellbinding approach to logic programs, *Proc. of the 1988 International Conference and Symposium on Logic Programming*, 1988, pp. 140--159

[17] Sato, T., and Tamaki, H. Enumeration of success patterns in logic programs. *Theoretical Computer Science*, vol. 34, pp. 227--240, 1984.

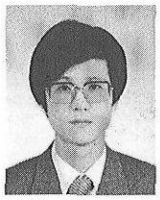
김 문 정

1993년 한남대학교 전자계산공학과 졸업(학사). 1995년 한국과학기술원 전산과 졸업(석사). 현재 (주)퓨쳐시스템 정보통신연구소 근무.

Copyright (C) 2005 NuriMedia Co., Ltd.

창 병 모

1988년 서울대학교 컴퓨터공학과 졸업(학사). 1990년 한국과학기술원 전산학과에서 공학석사 학위취득. 1994년 한국과학기술원 전산학과에서 공학박사 학위취득. 1994년 ~ 1995년 한국전자통신연구소 박사후 연수 연구원. 1995년 ~ 현재 숙명여자대학교 전산학과 조교수. 관심분야는 컴파일러 구성론(정적 분석, 코드 최적화), 논리 프로그래밍, 연역 데이타베이스.



최 광 무

1976년 서울대학교 전자공학과 졸업(학사). 1978년 한국과학기술원 전산학과 졸업(석사). 1984년 한국과학기술원 전산학과 졸업(박사). 현재 한국과학기술원 전산학과 부교수. 관심분야는 프로그래밍언어론, 논리 프로그램의 병렬 수행 및 컴파일러 구성임.

Copyright (C) 2005 NuriMedia Co., Ltd.