

이진 기수 조건에서 인접성 표현을 위한 최적화된 CNF 변환

(Optimal CNF Encoding for Representing Adjacency in Boolean Cardinality Constraints)

박 사 천 [†] 권 기 현 ^{**}
(Sachoun Park) (Gihwon Kwon)

요 약 만족가능성 처리기는 모델 검증 및 임베디드 프로그램 검증과 같이 소프트웨어 공학의 여러 분야에서 활용되고 있다. 만족가능성 처리기를 활용하기 위해서는 주어진 문제를 처리기의 입력인 CNF 형식으로 변환해야 한다. 그러나 이 형식은 소스코드나 소프트웨어 모델보다 표현력이 낮기 때문에 최적화된 변환이 요구된다. 본 논문에서는 “ n 개에서 인접된 $k \leq n$ 개 선택” 문제를 CNF형식으로 변환하는 최적화된 기법을 제시한다. 제안된 방법을 다양한 일본 퍼즐에 적용한 결과 우수한 성능이 입증되었다. 우리가 알고 있는 한, 인접성에 대한 최적화된 CNF 변환 연구는 거의 없다.

키워드 : 만족가능성 검사, 이진 기수 조건, 인접성

Abstract In some applications of software engineering such as the verification of software model or embedded program, SAT solver is used. To practical use a SAT solver, a problem is encoded to a CNF formula, but because the formula has lower expressiveness than software models or source codes, optimal CNF encoding is required. In this paper, we propose optimal encoding techniques for the problem of “Selecting adjacent $k \leq n$ among n objects.” Through experimental results we show the proposed constraint is efficient and correct to solve Japanese puzzle. As we know, this paper is the first study about CNF encoding for adjacency in BCC.

Key words : Satisfiability Checking, Boolean Cardinality Constraint, Adjacency

1. 서 론

모델 검증(Model checking)은 소프트웨어 모델 및 코드의 철저한 분석에 널리 사용되는 기술이다[1-4]. 모델 검증 방법에는 명시적인 방법과 기호적인 방법이 있다. 전통적으로 기호적인 모델 검증은 BDD(Binary De-

cision Diagram)를 기반으로 하는데, 이것은 상태폭발에 매우 취약한 것으로 판명되었고 그 결과 다룰 수 있는 시스템의 크기가 매우 제한적이다. 이러한 문제를 극복하기 위해서 BDD 대신 만족가능성 처리기[5,6]를 사용하는 연구가 많이 수행되고 있다. 바운드드(Bounded) 모델 검증은 일정 길이의 반례 찾는 것을 목표로 하는데 이때 만족가능성 처리기를 사용한다[7]. 그리고 시스템의 구조를 표현하고 검증할 때 사용되는 모델 검증기인 Alloy[8] 또한 만족가능성 처리기를 사용한다. 이러한 바운드드 모델 검증에서 핵심적인 문제는 모델을 CNF(Conjunctive Normal Form)로 변환하는 것이다.

모델을 만족가능성 문제로 변환하려면, 모델을 몇 가지 제약조건으로 표현하고 이를 CNF로 변환해야 한다. 많이 사용되는 제약조건 중에 이진 기수 조건(Boolean cardinality constraint)이 있다[9,10]. 이진 기수 조건은 주어진 n 개의 이진변수 중에서 참인 변수의 수를 $0 \leq k \leq n$ 으로 제약하는 조건이다. 이것은 n 개의 원소 중에서 k 개를 선택하는 조합의 문제로 볼 수 있다. 지

· 본 연구는 경기도의 경기도지역협력연구센터사업의 일환으로 수행하였음 (2008-111-11, 웹 기반 소프트웨어 모델 체크 도구 개발)

† 학생회원 : 경기대학교 컴퓨터과학과
sachem@kgu.ac.kr

** 정 회 원 : 경기대학교 컴퓨터과학과 교수
khkwon@kgu.ac.kr

논문접수 : 2008년 4월 14일
심사완료 : 2008년 9월 29일

Copyright©2008 한국정보과학회 : 개인 목적이나 교육 목적인 경우, 이 저작물의 전체 또는 일부에 대한 복사본 혹은 디지털 사본의 제작을 허가합니다. 이 때, 사본은 상업적 수단으로 사용할 수 없으며 첫 페이지에 본 문구와 출처를 반드시 명시해야 합니다. 이 외의 목적으로 복제, 배포, 출판, 전송 등 모든 유형의 사용행위를 하는 경우에 대하여는 사전에 허가를 얻고 비용을 지불해야 합니다.

정보과학회논문지: 소프트웨어 및 응용 제35권 제11호(2008.11)

금까지 이진 기수 조건을 CNF로 변환하는 연구들은 대부분 효율적인 CNF 생성에 초점을 맞춰 왔다. 여기서 효율성이란 첫째, 생성된 논리식이 주어진 변수 수에 대비해서 적은 수의 변수와 절을 갖도록 하는 것과 둘째, 생성된 CNF논리식이 만족가능성 처리기에서 보다 짧은 시간에 처리되는 것을 의미한다. 본 논문에서는 문제를 CNF로 변환할 때 이진 기수 조건과 함께 사용할 수 있는 인접성에 대한 개념을 소개하고 최적화된 CNF 변환 방법을 제안한다. 인접성은 주어진 이진 변수들이 일정한 순서를 갖는다고 가정한다. 그리고 선택된 변수들이 모두 이웃했을 때, 인접한다고 한다.

[9]에서는 이진 기수 조건을 CNF로 변환하는데 이진 트리 구조를 이용했고, 제안한 변환 방법의 효율성을 보이기 위해서 이산 단층촬영 문제를 다루었다. 또한 [11]에서는 일본 퍼즐[12]을 풀이하는데 진화 알고리즘을 적용한 사례를 보였다. 그러나 문제를 이산 단층촬영 문제와 동일하게 취급했기 때문에 25×25 크기의 비교적 작은 예제를 복원하는 데에도 12시간의 시간이 걸렸다. 우리는 일본 퍼즐을 해결하기 위해서 인접성을 고려한 정보를 사용했다. 실험은 세 종류의 데이터 즉, 블록면체 형태의 데이터, 무작위 데이터, 그리고 일본 퍼즐 예제를 가지고 수행했다. 본 논문에서는 네 가지의 변환 방법으로 주어진 데이터에 대해서 각각 실험했다. 두 가지는 기존의 방법에 인접성을 추가한 것이고 다른 두 가지는 인접성을 적용하기 위해 최적화된 방법을 사용했다. 실험을 통해서 최적화된 방법이 적은 절 수와 변수 수를 가지며, 더 빨리 처리되는 것을 확인할 수 있었다.

본 논문의 구성은 다음과 같다. 2장에서는 배경 지식으로 이진 제약 조건의 CNF 변환을 설명한다. 여기서는 이진 제약 조건을 이진 변수의 집합 E 에서 참인 값을 갖는 변수의 집합이 E' 이라고 할 때, 집합의 크기를 $l \leq |E'| \leq m$ 으로 제약하는 것으로 설명한다. 일반적인 변환방법, 이진 트리를 이용한 방법, 그리고 순차 계수기를 이용한 방법 등을 소개하고 그 특징들을 설명한다. 3장에서는 인접성의 개념을 소개하고 그 변환 방법에 대해서 일반적인 방법, 이진 트리를 이용한 방법, 그리고 순차 계수기를 이용한 방법을 제안한다. 4장에서는 3장에서 제안된 방법을 최적화하는 기법을 다루고, 5장에서는 일본 퍼즐 풀이로써 인접성을 표현한 기법들에 대한 비교 실험을 하고, 6장에서는 결론을 맺는다.

2. 배경 지식

이번 장에서는 이진 제약 조건을 CNF식으로 변환하는 기존의 방법을 설명한다. 먼저 Naïve 변환 방식과 이 변환의 문제점을 살펴본 후 효율성을 고려한 두 가지 변환 방식을 소개한다.

2.1 Naïve 변환

본 절에서는 이진 기수 조건을 나타내는 일반적인 방법을 설명한다. n 개의 원소를 갖는 이진 변수의 집합 $E = \{e_1, \dots, e_n\}$ 이 주어졌을 때, 참인 값을 갖는 변수들의 부분 집합을 $E' \subseteq E$ 라고 하자. 이 부분 집합 E' 에 대한 기수 조건 $l \leq |E'| \leq m$ 을 CNF로 변환해 보자. 이것은 다시 두 조건 $l \leq |E'|$ 와 $|E'| \leq m$ 이 동시에 만족하는 것으로 대신할 수 있다.

조건 $l \leq |E'|$ 과 같이 집합의 크기가 l 보다 크거나 같다는 것은 최소한 l 이라는 것이다. 따라서 이 조건을 이진 변수로 나타내기 위해서 크기가 $n-l+1$ 인 E 의 모든 부분 집합 $E'_1, \dots, E'_{\binom{n}{l-1}}$ 에 대해서, 아래 식으로 표현된다[13].

$$\bigwedge_{i=1}^{\binom{n}{l-1}} \bigvee_{j=1}^{n-l+1} e_j \in E'_i \quad (1)$$

예를 들어서 5개의 이진 변수 중에 최소한 2개가 참인 경우를 생각해 보자. 이때, 이진 변수의 집합은 $E = \{e_1, \dots, e_5\}$ 이고 크기가 $n-l+1=4$ 인 E 의 모든 부분 집합은 $\binom{n}{l-1}=5$ 개이다. 이런 식으로 각각의 부분 집합 E'_1, \dots, E'_5 를 구하면 아래와 같다.

$$E'_1 = \{e_1, e_2, e_3, e_4\}$$

$$E'_2 = \{e_1, e_2, e_3, e_5\}$$

$$E'_3 = \{e_1, e_2, e_4, e_5\}$$

$$E'_4 = \{e_1, e_3, e_4, e_5\}$$

$$E'_5 = \{e_2, e_3, e_4, e_5\}$$

이제 (1)번 식을 나타내면, 길이가 4인 각 부분 집합 내의 변수들은 논리합으로 연결되고 각각의 집합들은 논리곱으로 묶이는 하나의 CNF 식이 생성된다. 이 식에서 만일 e_1 이 참이라고 가정해 보자. 이때 맨 마지막 절에는 e_1 이 존재하지 않으므로 $\{e_2, e_3, e_4, e_5\}$ 중에 적어도 하나는 참이 되어야 하고 이것은 논리합으로 연결되기 때문에 반드시 만족하게 된다. 따라서 최소한 두 개의 변수는 참이 된다. 위의 각 부분집합은 전체 집합에 비해서 원소의 개수가 하나씩 부족하기 때문에 어떤 변수를 참으로 가정하더라도 최소한 두 개의 변수가 참이 된다는 것을 확인할 수 있다.

이제 두 번째 조건 $|E'| \leq m$ 을 CNF로 변환해 보자. 집합의 크기가 m 보다 작거나 같다는 것은 최대 m 이라는 것이다. 따라서 크기가 m 인 E 의 모든 부분집합 E' 에 대해서 “ E' 에 속한 원소가 모두 참이라면, $E-E'$ 에 속한 원소들은 모두 거짓”이라는 방식으로 기술한다. $|E'| \leq m$ 은 크기가 $m+1$ 인 E 의 모든 부분 집합

$E''_1, \dots, E''_{\binom{n}{m+1}}$ 에 대해서, 아래 식으로 표현할 수 있다[10].

$$\bigwedge_{i=1}^{\binom{n}{m+1}} \bigvee_{j=1}^{m+1} \neg e_j \in E''_i \quad (2)$$

이진 변수 5개 중에 최대한 2개를 선택하는 예를 위의 수식으로 표현해 보자. 먼저 크기가 3인 E 의 부분 집합의 개수는 10개이다. 이러한 부분집합들을 E''_1, \dots, E''_{10} 라고 하면 아래와 같이 나열할 수 있다.

$$\begin{aligned} E''_1 &= \{e_1, e_2, e_3\} \\ E''_2 &= \{e_1, e_2, e_4\} \\ E''_3 &= \{e_1, e_2, e_5\} \\ E''_4 &= \{e_1, e_3, e_4\} \\ E''_5 &= \{e_1, e_3, e_5\} \\ E''_6 &= \{e_1, e_4, e_5\} \\ E''_7 &= \{e_2, e_3, e_4\} \\ E''_8 &= \{e_2, e_3, e_5\} \\ E''_9 &= \{e_2, e_4, e_5\} \\ E''_{10} &= \{e_3, e_4, e_5\} \end{aligned}$$

이제 (2) 번식을 나타내면, 위의 10개의 부분 집합의 원소들은 각각 부정기호(\neg)가 붙은 논리합으로 연결된다. 즉 CNF의 하나의 절이 된다. 그리고 각 절들은 논리곱으로 연결된다. 즉, 아래 같은 식이 된다.

$$\begin{aligned} &(\neg e_1 \vee \neg e_2 \vee \neg e_3) \wedge (\neg e_1 \vee \neg e_2 \vee \neg e_4) \wedge \dots \wedge \\ &(\neg e_3 \vee \neg e_4 \vee \neg e_5) \end{aligned}$$

위 식에서 만일 e_1 과 e_2 가 참인 변수라고 가정해 보자. 그렇다면 위 식에 의해서 e_3, e_4, e_5 는 모두 거짓이 된다. 왜냐하면 위 식의 첫 번째 절 $(\neg e_1 \vee \neg e_2 \vee \neg e_3)$ 은 $(e_1 \wedge e_2 \Rightarrow \neg e_3)$ 과 동치인데 첫 번째부터 세 번째까지의 절이 만족하려면 e_3, e_4, e_5 는 모두 거짓이 되어야 하기 때문이다. 다른 변수를 선택하더라도 결과는 이와 같다.

살펴본 바와 같이 이진 변수의 집합 $E = \{e_1, \dots, e_n\}$ 에서 k 개의 원소를 선택하는 것은 크기가 k 인 부분 집합 $E' \subseteq E$ 에 대해서 조건 $k \leq |E'| \leq k$ 를 위와 같은 식으로 변환함으로써 나타낼 수 있다. 그런데 $|E'| \leq k$ 에 대한 변환을 통해서 $(k+1)$ 의 길이를 갖는 $\binom{n}{k+1}$ 개의 절이 생성된다. 또한 최악의 경우 $k = \lceil n/2 \rceil - 1$ 일 때, $O(2^n / \sqrt{n/2})$ 개의 절이 생성된다. 다음 절에서는 조건 $l \leq |E'| \leq m$ 을 표현하는데 보다 적은 수의 절을 가지고 변환하는 방법을 소개한다.

2.2 이진 트리구조를 이용한 변환

앞 절에서는 기수 조건 $l \leq |E'| \leq m$ 을 표현하기 위해, 조건을 다시 $l \leq |E'|$ 와 $|E'| \leq m$ 으로 나누고 이것들을

직접 CNF로 변환했다. 본 절에서는 이진 변수의 집합 $E = \{e_1, \dots, e_n\}$ 에서 참인 값을 갖는 이진 변수의 개수를 나타내기 위해서 추가적인 변수들을 사용한다. 여기에서는 정수를 이진 변수로 나타낼 때, 단항(unary) 표현을 사용한다[9].

원소의 개수가 5인 이진 변수의 집합 $E = \{e_1, \dots, e_5\}$ 에서 참인 변수의 수를 단항 표현으로 나타내려면 5개의 보조 변수가 필요하다. 추가적인 보조 변수의 집합을 $S = \{s_1, \dots, s_5\}$ 라고 하자. 만일 E 에서 참인 변수가 e_1 와 e_4 둘뿐이라면 집합 S 에 의한 단항 표현은 $s_1 \wedge s_2 \wedge \neg s_3 \wedge \neg s_4 \wedge \neg s_5$ 가 된다. 따라서 이진 변수 s_1, s_2, \dots, s_n 로 정수 $0 \leq k \leq n$ 을 나타낼 때, 단항 표현을 사용하면 $s_1 \wedge \dots \wedge s_k \wedge \neg s_{k+1} \wedge \dots \wedge \neg s_n$ 이 된다.

이진 변수의 집합 $S = \{s_1, \dots, s_n\}$ 으로 범위 $l \leq k \leq m$ 의 단항 표현을 나타내면 아래와 같다.

$$\bigwedge_{i=1}^l s_i \wedge \bigwedge_{j=m+1}^n \neg s_j \quad (3)$$

이와 같이 임의의 이진 변수의 집합 E 에서 선택할 수 있는 모든 경우를 S 의 단항 표현으로 나타내기 위해서 [9]는 트리 구조를 사용했다. 그림 1(a)를 보면, 트리의 비 단말 노드에 보조 변수 l_1, l_2 와 r_1, r_2, r_3 그리고 r_4, r_5 가 각각 배정되고 루트 노드에는 S 를 배정하고 트리의 단말 노드에 E 의 원소가 하나씩 배정된다. 따라서 모든 노드들은 각자 자신의 하위 트리에서 선택된 개수의 합을 나타내는 단항 표현이라고 할 수 있다. 따라서 앞의 예에서처럼 참인 변수가 e_1, e_4 뿐이라면, 각 중간 노드의 변수들은 한 개씩 선택됨을 나타내기 위해 각각 $l_1 \wedge \neg l_2$ 와 $r_4 \wedge \neg r_5$ 그리고 $r_1 \wedge \neg r_2 \wedge \neg r_3$ 가 되어야 한다.

결론적으로 이진 트리 구조를 이용해서 조건 $l \leq |E'| \leq m$ 을 표현하기 위해서는 그림 1(b)와 같이 먼저 전체 원소의 집합 $E = \{e_1, \dots, e_n\}$ 에 대한 이진 트리를 만들어야 한다. 앞의 예에서처럼 이 트리의 각 단말 노드에는 E 의 원소가 하나씩 배정된다. 그리고 트리의 모든 비 단말 노드에는 해당 자식 노드들의 합을 단항 표현으로 나타내기 위한 추가적인 보조변수의 집합이 배정된다. 마지막으로 루트 노드에는 E 에서 참인 변수의 합을 표현하기 위한 보조변수 $S = \{s_1, \dots, s_n\}$ 가 배정된다. 또한 트리의 균형을 맞추기 위해서 루트의 왼쪽 자식 노드에는 $\lfloor \frac{n}{2} \rfloor$ 개의 이진 변수를, 오른쪽 자식 노드에는 $n -$

$\lfloor \frac{n}{2} \rfloor$ 개의 이진 변수를 배정한다. 따라서 그림 1(b)에서 $m_1 = \lfloor \frac{n}{2} \rfloor$ 이고 $m_2 = n - \lfloor \frac{n}{2} \rfloor$ 이며 $m_1 + m_2 = n$

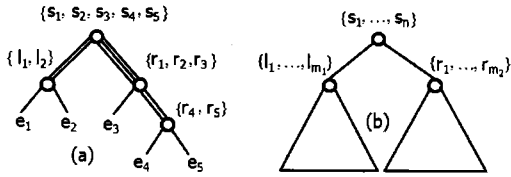


그림 1 단항 표현을 위한 이진 트리

이다. 이와 같은 방식을 모든 서브 트리에 재귀적으로 적용해서 전체 트리를 완성할 수 있다.

이진 트리에서 임의의 비 단말 노드에 이진 변수 집합 $A = \{a_1, \dots, a_p\}$ 가 배정되고 그의 왼쪽과 오른쪽 자식 노드에는 각각 $L = \{l_1, \dots, l_{p_1}\}$ 와 $R = \{r_1, \dots, r_{p_2}\}$ 가 배정되었다고 하자. 이들간의 관계는 다음과 같이 CNF로 변환된다. 이때, $a_0 = l_0 = r_0 = 1$ 이고 $a_{p+1} = l_{p+1} = r_{p+1} = 0$ 이다.

$$\bigwedge_{i=0}^{p_1} \bigwedge_{j=0}^{p_2} (l_i \wedge r_j \Rightarrow a_{i+j}) \wedge (a_{i+j} \Rightarrow l_{i+1} \vee r_{j+1}) \quad (4)$$

모든 비 단말 노드에 대해서 위와 같은 식으로 나타내면 전체 트리 구조를 표현한 CNF 식을 얻게 된다. 따라서 조건 $l \leq |E'| \leq m$ 을 변환하는 것은 (3)과 (4)의 논리곱으로 나타낼 수 있다. 이진 트리를 이용한 변환에는 $O(n \log n)$ 개의 추가 변수들과 $O(n^2)$ 개의 절이 필요하다. 다음 절에서는 이진 기수 조건을 순차 계수기를 사용하는 방법을 살펴본다.

2.3 순차 계수기를 이용한 변환

본 절에서는 순차 계수기를 사용하는 변환 방법을 설명한다. 순차 계수기는 그림 2(a)와 같이 이진 입력 변수 e_1, \dots, e_n 을 차례로 더해서 그 합을 단항 표현으로 나타낸다. 그런데 이 서킷에서 계산할 수 있는 최대값은 m 이 된다.

다. 그림 2(b)는 i 번째 부분합 $s_i = \sum_{j=1}^i e_j$ 를 $s_{i,1}, \dots, s_{i,m}$ 의 단항 표현으로 나타내는 서킷이다. 따라서 덧셈을 위해서

$(n-1) \times m$ 개의 추가적인 이진 변수가 필요하다[10].

조건 $|E'| \leq m$ 를 의미하는 그림 2의 순차 계수기를 CNF 논리식으로 변환하면 다음과 같다.

$$\begin{aligned} & e_1 \Rightarrow s_{1,1}^* \\ & \text{for } 1 < j \leq m \\ & \quad \neg s_{1,j} \\ & \text{for } 1 < i < n \\ & \quad (e_i \vee s_{i-1,1}) \Rightarrow s_{i,1}^* \quad (5) \\ & \quad \text{for } 1 < j \leq m \\ & \quad \quad ((e_i \wedge s_{i-1,j-1}) \vee s_{i-1,j}) \Rightarrow s_{i,j}^* \\ & \quad e_i \Rightarrow \neg s_{i-1,m} \\ & \quad e_n \Rightarrow \neg s_{n-1,m} \end{aligned}$$

[10]에서는 위의 서킷을 CNF로 변환하기 위해서 식 (5)와 같이 함축(implication)기호를 사용했다. 이러한 방법은 이진 변수들의 합이 최대한 m 라는 것, 즉 조건 $|E'| \leq m$ 에 대한 변환을 최적화 할 수 있다. 그러나 *표시가 된 논리식에서 함축 기호 대신에 동등(equivalence)기호를 사용하면, 적은 절수를 유지하면서도 조건 $l \leq |E'| \leq m$ 을 표현할 수 있다.

(5)의 *부분에 동등 기호를 사용했다고 가정하고 조건 $l \leq |E'|$ 을 순차 계수기를 이용해서 변환하면 아래와 같다.

$$s_{i,l} \vee \dots \vee s_{n-1,l} \vee (e_n \wedge s_{n-1,l-1}) \quad (6)$$

$l \leq |E'| \leq m$ 를 CNF식으로 변환할 때, 동등 기호를 사용한 (5)와 (6)의 논리곱을 사용하면 $4mn - 7m + 4$ 개의 절이 생성된다. 이것은 $|E'| \leq m$ 을 변환하기 위해서 함축 기호를 사용한 절의 수 $2mn + n - 3m - 1$ 보다는 많지만, $l \leq |E'|$ 부분을 표현하기 쉬웠고, 절수의 증가 비율도 선형적이다.

3. 인접성의 CNF 변환

인접성이란 n 개의 이진 변수 중에 k 개를 선택하는데

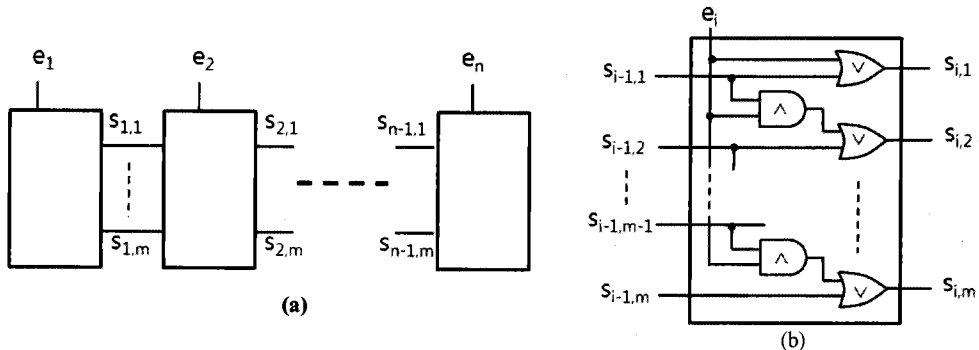


그림 2 순차 계수기

임의의 k 개가 아닌 서로 인접한 이진 변수들을 선택한다는 의미이다. 따라서 각각의 이진 변수들은 고유의 위치가 주어져야 하고 그들의 순서가 고려되어야 한다. 따라서 지금까지는 순서 없는 이진변수의 집합 E 를 다루었는데, 이번 장에서는 E 를 시퀀스로 간주한다. 시퀀스 E 에서 길이가 k 인 서브시퀀스 E' 을 추출하는 문제는 2장의 각 절에서 기술했던 이진 서수 조건에 각각의 인접성 조건을 추가하는 방식으로 변환 한다.

3.1 인접성에 대한 Naive 변환

인접성을 나타내기 위해서는 변수들 간의 순서가 있음을 가정해야 한다. 예를 들어 순서가 있는 5개의 이진 변수 $\{e_1, e_2, e_3, e_4, e_5\}$ 중에 인접한 2개의 변수가 참이 되는 것을 표현해보자. 만일 e_3 변수가 참이 된다면, e_2 혹은 e_4 변수가 참이 되어야 한다. 이런 방식으로 가능한 경우를 모두 표현해 보면 “ $(e_1 \wedge e_2), (e_2 \wedge e_3), (e_3 \wedge e_4), (e_4 \wedge e_5)$ 중에 정확하게 하나만 참이 된다.”라고 할 수 있다. 이것을 긍정표현이라고 하자. 이것을 CNF 식으로 변환하기 위해서 우리는 보조변수 a 를 사용해서 $a \Leftrightarrow (e_1 \wedge e_2)$ 와 같은 형태로 나타낸다. 그리고 그 보조변수 중에 정확히 하나만 참이라는 제약사항을 추가하면 CNF 변환을 완성할 수 있다. n 개의 이진 변수의 집합 $E = \{e_1, \dots, e_n\}$ 에서 인접한 k 개의 변수를 선택하는 문제는 아래와 같이 변환된다.

$$\bigwedge_{i=1}^{n-k+1} \left(a_i \Leftrightarrow \bigwedge_{j=i}^{i+k-1} e_j \right) \wedge \text{Exactly1}(\{a_1, \dots, a_{n-k+1}\}) \quad (7)$$

여기서 $\text{Exactly1}(\cdot)$ 은 이진 변수의 집합을 받아서 해당 집합에서 정확하게 하나의 변수만 참이 되는 조건을 CNF 변환한 식을 돌려주는 함수이다. 이때 $n-k+1$ 개의 보조변수와 $(n-k+1)(k+1)+\alpha$ 개의 절이 필요하다. 여기서 α 는 보조변수 중 정확히 하나만 참이라는 조건을 표현하는데 사용된 절의 수로써 본 논문에서는 정확히 하나를 선택할 때 지금까지 발표된 논문 중에 가장 좋은 결과를 보이는 [14]의 변환 방법을 사용한다.

앞의 예에서 인접성에 대한 또 다른 관찰은 “ e_3 변수가 참이 된다면, e_1 과 e_5 변수는 반드시 거짓이어야 한다.”라는 것이다. 이러한 방식으로 모든 변수의 관계를 논리식으로 나타내면, $(e_1 \Rightarrow \neg e_3 \wedge \neg e_4 \wedge \neg e_5) \wedge (e_2 \Rightarrow \neg e_4 \wedge \neg e_5) \wedge (e_3 \Rightarrow \neg e_5)$ 이 된다. 이러한 방식을 부정표현이라고 하자. 따라서 이진 변수의 집합 $E = \{e_1, \dots, e_n\}$ 에 대해서 n 개 중에 인접한 k 개의 선택하는 부정적인 표현으로 일반화하면, $\frac{(n-k+1)(n-k)}{2}$ 개의 절이 요구되고, 아래와 같이 표현할 수 있다.

$$\bigwedge_{i=1}^{n-k} \bigwedge_{j=i+k}^n e_i \Rightarrow \neg e_j \quad (8)$$

이렇게 인접성을 나타내는 것 자체로 인접한 k 개의 변수가 참이 되도록 할 수는 없다. 앞 장에서 다루었던 기수 조건을 함께 사용해야 한다. 즉 이진 변수의 기수 조건은 선택될 개수를 제약하고 인접성 조건은 선택된 변수들이 인접한다는 것을 제약한다. 그런데, 인접성의 긍정표현은 $k \leq |E'|$ 에 포함된다. 왜냐하면 인접한 k 개는 $\binom{n}{k}$ 에 포함되기 때문이다. 따라서 긍정표현은 $|E'| \leq k$ 조건과 논리곱 해서 변환하면, 정확히 k 개의 인접한 변수들이 참이 되는 것을 표현할 수 있다. 또한 부정표현은 어떤 변수가 선택 될 때, $(n-k)$ 개의 변수들이 거짓이 되도록 표현하므로 $|E'| \leq k$ 에 포함된다. 따라서 $k \leq |E'|$ 조건과 논리곱 해서 변환해야 인접성을 완성할 수 있다. 다음절에서는 앞 장에서 다루었던 두 가지 효율적인 변환 방법에 본 절에서 다루었던 인접성의 부정표현을 추가하는 방법에 대해서 기술한다.

3.2 이진 트리 방식에 인접성 추가

인접성의 부정표현을 이진 트리 구조해서 효율적으로 나타내려면 비 단말 노드에 배정된 변수가 단말 노드의 이진 변수들의 진위 값을 조절할 수 있다는 사실에 주목해야 한다.

2.2절의 그림 1(a)는 예제 1을 나타내는 그림인데, 이를 이용해서 인접성의 부정표현을 나타내면 $(e_1 \Rightarrow \neg r_1) \wedge (e_2 \Rightarrow \neg r_4) \wedge (e_3 \Rightarrow \neg r_5)$ 가 된다. 여기서 $\neg r_1$ 은 $\neg e_3 \wedge \neg e_4 \wedge \neg e_5$ 를 나타내고 $\neg r_4$ 은 $\neg e_4 \wedge \neg e_5$ 를 나타낸다.

이진 변수의 집합 $E = \{e_1, \dots, e_n\}$ 이 주어졌고 그림 1-(b)와 같이 이진 기수 조건을 위한 이진 트리가 구성되었다면, 이에 대한 인접성은 아래와 같은 식으로 표현 된다.

$$\bigwedge_{i=1}^{n-k} \bigwedge_{a \in LCA(i)} e_i \Rightarrow \neg r_1^{\text{right}(a)} \quad (9)$$

여기서 $LCA(i) = \{lca(i, j) \mid i+k \leq j \leq n\}$ 을 의미하고, N 을 자연수의 집합이라고 할 때, 함수 $lca: N \times N \rightarrow N$ 는 두 단말노드의 노드번호를 받아서 두 노드의 최소 공통 조상 노드의 노드 번호를 돌려주는 함수이다. 최소 공통 조상 노드는 두 노드의 공통 조상이면서 깊이가 가장 깊은 노드이다. 또한 $right: N \rightarrow N$ 는 비 단말 노드의 노드 번호를 입력으로 받아서 그 노드의 오른쪽 자식 노드의 노드 번호를 돌려주는 함수이다. 따라서 $r_1^{\text{right}(a)}$ 은 a 의 오른쪽 자식 노드에 첫 번째 배정된 변수를 의미한다.

식 (9)를 (3)과 (4)와 함께 논리 곱하면 이진 트리를

이용한 인접성 변환이 완성된다. (9)의 변환 방식은 부정형태 이므로 추가적인 변수가 요구되지 않고, 트리를 이용하기 때문에 최악의 경우 $(n-k) \cdot \log n$ 개의 절이 사용된다. 이러한 방식을 TA라고 부를 것이다.

그런데, 앞서 언급했던 바와 같이 인접성에 대한 표현은 긍정표현과 부정표현이 있고, 긍정표현은 $|E'| \leq m$ 조건과 논리곱으로 인접성을 변환할 수 있고, 부정표현은 $l \leq |E'|$ 조건과 논리곱 함으로써 변환할 수 있다고 했다. 따라서 $l \leq |E'|$ 조건을 의미하는 (4)의 왼쪽 부분과 (8)의 식을 논리곱 한 것을 TL+NR이라고 하고, (4)에서 $|E'| \leq m$ 조건을 의미하는 오른쪽 부분과 (7)의 식을 논리곱 한 것을 TR+PR이라고 하면, 이진 트리를 이용한 인접성은 TA, TL+NR, 그리고 TR+PR 등 세 가지 방식이 가능하게 된다.

3.3 순차 계수기 방식에 인접성 추가

순차 계수기를 이용한 인접성도 세 가지 방식으로 표현 될 수 있다. 식 (5)를 순차 계수기(Sequential Counter)의 약자인 SC라고 하자. 앞서 언급한 바와 같이 SC는 조건 $|E'| \leq m$ 를 변환하는 것에 최적화 되어있다. 이것은 참인 변수의 수가 최대 m 개 라는 뜻인데, 만일 거짓인 변수가 최대 $(n-l)$ 개 라고 표현한다면 쉽게 조건 $l \leq |E'|$ 를 표현할 수 있다. 이를 위해서 SC에서 입력 변수들을 부정하고, 입력 l 에 대해서 $(n-l)$ 개의 변수를 선택하도록 하는 식을 SC^{-1} 이라고 하자. 그리고 (5)의 *표시 된 식에 함축 기호 대신 동등 기호를 사용해서 식 (6)과 논리곱 한 식을 SC^M 이라고 하자.

인접성을 표현하는 첫 번째 방식은 SC+PR이다. 즉 긍정표현과 $|E'| \leq m$ 을 논리곱 하는 것이다. 두 번째 방식은 SC^{-1} +NR이다. 이것은 조건 $l \leq |E'|$ 을 표현할 수 있도록 변형한 식 SC^{-1} 과 부정표현의 논리곱으로 이루어진다. 또한 SC^M 에서 계수기의 구조를 이용해서 인접성을 추가하는 방식이 있다. SC^M 은 동치 기호를 사용하기 때문에 부분함을 나타내는 변수로 입력 변수들의 진 위 값을 제약할 수 있다. 따라서 각각의 입력변수에 대해서 부정표현 방식으로 나타내면 아래의 식 (10)과 같다.

$$\bigwedge_{i=1}^{n-m} e_i \Rightarrow s_{i+m-1,m} \wedge \bigwedge_{i=m+1}^n e_i \Rightarrow \neg s_{i-m,1} \quad (10)$$

위의 왼쪽 부분은 그림 3의 아래에서 음영으로 표시한 오른쪽 부분을 나타내는 식이고 식의 오른쪽 부분은 그림에서 음영으로 표시한 왼쪽 부분을 나타내는 식이다. 따라서 추가적인 변수 없이 $2(n-m)$ 개의 절로 표현된다.

4. 인접성의 CNF 변환 최적화

3장에서는 이진 트리를 이용하는 이진 기수 조건에

인접성을 추가하는 세가지 방식과 순차 계수기를 이용하는 기수 조건에 인접성을 추가하는 세가지 방법을 살펴 보았다. 이번 장에서는 트리를 이용해서 인코딩을 최적화하는 방법과 순차 계수기를 적용해서 인코딩을 최적화하는 방법을 기술한다.

4.1 이진 트리구조에서 인접성 변환 최적화

앞서 언급한 바와 같이 (4)의 왼쪽 식과 (3)을 논리곱 하면 $l \leq |E'|$ 을 변환할 수 있는데, 이를 더 간략한 식으로 표현할 수 있다. 그림 3은 2.2절의 예제에 대한 식 (4)의 왼쪽부분을 전개한 것이다.

그런데 식 (3)은 단항 표현이기 때문에 모두 단일 절로 구성된다. 만일 전체 변수의 수가 n 이고, 선택하는 변수의 개수가 k 라면 각 비 단말 노드에 배정된 변수 중 $k < x \leq n$ 인 변수는 모두 제거될 수 있다. 따라서 단일 절로 표현되는 식은 CNF 변환할 때 만족가능성 처리기에 입력되기 전에 구문적으로 제거 할 수 있다. 그림 3에서 음영으로 표시된 부분은 모두 변환할 때 구문적으로 제거되는 부분이 된다. 다만 함축기호와 함께 제거 되기 때문에 앞의 식에 부정기호가 추가된다. 이렇게 해서 만들어진 식은 (3)의 의미대로 이진 트리를 이

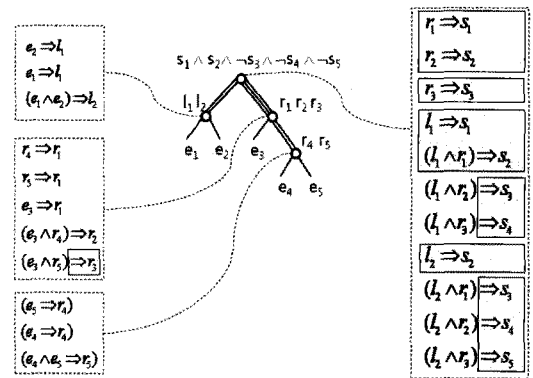


그림 3 2.2절의 예제에 대한 식 (4)의 $l \leq |E'|$ 부분

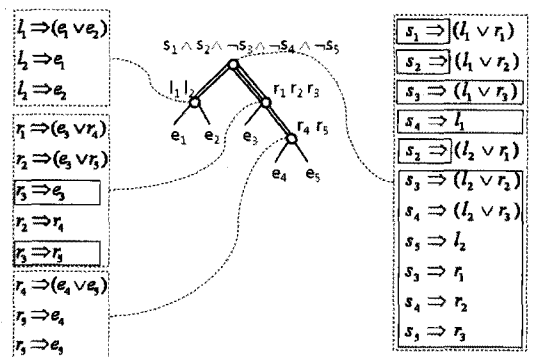


그림 4 2.2절의 예제에 대한 식 (4)의 $|E'| \leq m$ 부분

용해서 조건 $l \leq |E'|$ 을 최적화한 식이 된다.

그림 4는 식 (4)의 오른쪽 부분인 $|E'| \leq m$ 조건을 전개한 것이다. $l \leq |E'|$ 을 최적화한 것처럼 그림 4의 음영으로 표시된 부분은 주어진 k 에 의해서 구문적으로 제거될 수 있는 부분이다. 이렇게 최적화된 식에서는 최악의 경우 $k \cdot \sum_{i=1}^{\log(n-1)} 2^i$ 개의 추가 변수들과 $O(n^2)$ 개의 절이 생성된다. 최적화를 통해서 위의 예제에서는 17개의 변수 중에 6개의 변수가 줄었고 44개 중 18개의 절이 줄어들었다.

4.2 순차 계수기에서 인접성 변환 최적화

표 1은 n 개 중에 인접한 k 개의 선택하는 SC+PR 방식과 $SC^{-1}+NR$ 방식에 대한 간단한 비교 실험결과이다. 각각 절 수, 변수 수, 그리고 처리시간이다. 아래 표에서도 알 수 있듯이 순차 계수기를 이용한 변환의 변수 수와 절 수는 선택하는 개수 즉 k 에 의존적이다. 앞서 설명했던 것과 같이 덧셈을 위한 $(n-1) \cdot k$ 개의 보조변수가 필요하기 때문이다.

또한 긍정표현인 (7)의 인접성은 k 의 크기가 $\frac{n}{2}$ 일 때, 가장 많은 절 수를 만들어 내고, 부정표현인 식 정의 (8)의 인접성은 k 의 크기가 클수록 더 적은 절수로 변환된다. 예상했던 결과이기도 하지만, 흥미로운 것은 SC+PR 방식은 k 가 증가함에 따라 절 수와 변수 수가 증가하고 처리시간도 더 오래 걸렸던 것에 반해서 $SC^{-1}+NR$ 방식은 그 반대의 결과를 보여주었다. 이것은 SC^{-1} 에 k 를 입력하면 최대 $(n-k)$ 개의 변수가 거짓이라는 식을 만들기 때문이다. 따라서 $k < \frac{n}{2}$ 이면

SC+PR의 방식이 효율적이고, 반대로 $\frac{n}{2} \leq k$ 이면 $SC^{-1}+NR$ 방식이 효율적이다. 따라서 이 두 방식을 선택해서 k 값에 따라 변환하는 방식을 CS라고 부를 것이다. 따라서 CS 방식은 크기가 작은 CNF 논리식을 만들어 내며 순차 계수기를 이용해서 인접성을 나타내는 최적화된 방식이 된다.

표 1 $SC^{-1}+NR$ 과 SC+PR의 비교

n/k	$SC^{-1}+NR$			SC+PR		
	#변수	#절	처리시간	#변수	#절	처리시간
100/20	2200	6017	0.031	8020	19099	0.062
100/50	5127	12725	0.062	5050	11224	0.031
100/80	8053	17630	0.093	2080	4249	0.031

5. 사례 연구

지금까지 다양한 기수제약 조건의 CNF 변환에 대해

서 살펴봤고, 인접성에 대한 정의와 그에 대한 변환 방법에 대해서 설명했다. 그리고 인접성을 나타내는 최적화된 방법들을 생각해 보았다. 이번 장에서는 최적화된 방법이 얼마나 효율적인지 실험을 통해서 알아보겠다.

5.1 적용 분야 및 방법

[9]에서는 제안한 CNF 변환의 효율성을 보이기 위해 네 방향에서 단층촬영(Discrete Tomography)한 투영 데이터를 이진 기수 조건을 써서 CNF로 변환하고 만족 가능성 처리기로 처리한 후 그 결과로부터 원래의 이미지를 얻어내었다. 본 논문에서는 이미지가 그림 5와 같이 일본 퍼즐의 형태로 주어진다고 가정한다. 일본 퍼즐은 $n \times m$ 의 빈 칸이 주어지고, 각 행과 각 열에 검은색 박스가 연속해서 나타나야 할 숫자들이 주어지면, 주어진 숫자의 개수만큼 빈 칸을 채워나가는 퍼즐이다. 단, 각 숫자에 해당하는 연속된 검은 박스들은 하나 이상의 흰색 박스에 의해서 구별되어야 한다. 위 그림의 데이터로부터 원래의 이미지를 만들기 위해, 우리는 [9]에서 사용했던 방법과 같이 만족가능성 처리기를 사용한다. 그러나 [9]와는 다르게 본 논문에서는 3장과 4장에서 제안한 인접성을 사용해서 문제를 변환한다.

변환 방법을 요약하면 다음과 같다. 각 칸에는 하나의 이진 변수를 배치한다. 그리고 각 행과 열에 인접성을 적용하기 위해서, 주어진 숫자 만큼 인접성 변환을 수행한다. 예를 들어, 그림 5에서 마지막 행에 배정된 이진 변수들이 e_1, \dots, e_{20} 라고 하자. 주어진 2,1이라는 숫자에 대해서, 첫 번째 숫자 2가 표시될 수 있는 가장 큰 공간은 첫 18칸 즉, e_1, \dots, e_{18} 이고 두 번째 숫자 1이 표시될

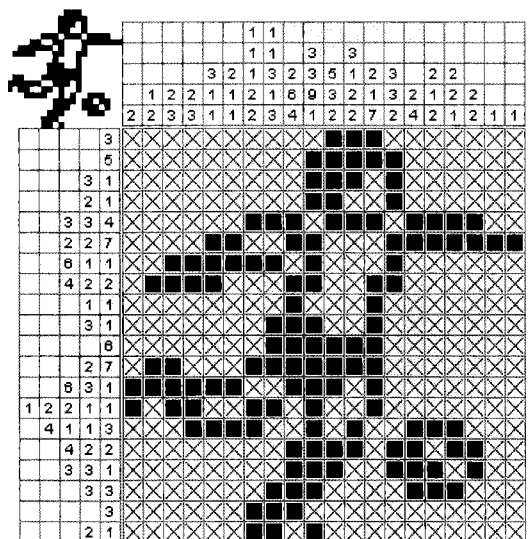


그림 5 20x20 일본 퍼즐의 예

수 있는 가장 큰 공간이 4번째 칸부터 마지막 칸까지 즉, e_4, \dots, e_{20} 이다.

따라서 18개의 이진 변수 중에 인접한 2개를 선택하는 것과 17개의 이진 변수 중에 1개를 선택하는 두 개의 문제로 나뉘어진다. 이때, 이진변수들 e_4, \dots, e_{18} 이 중복되어 변환된다. 이러한 중복은 엉뚱한 결과를 야기하므로 중복을 피하기 위해서 중복된 셀에는 새로운 이진 변수를 배정하는 방식으로 변환한다. 이때 간섭이 발생하는 셀에 대한 변환을 추가로 수행하고, 가로 세로 칸에 대한 변수를 일치시키면 전체 변환이 완성된다.

5.2 실험결과 및 분석

실험을 통해서 확인하고자 하는 바는 첫째, 그림 6과 같은 정보가 주어지면, 인접성 변환을 활용해서 문제를 해결할 수 있다는 것과 둘째, 최적화하지 않은 방법보다는 최적화 된 방법이 더 적은 절 수와 변수 수를 갖도록 CNF로 변환되며 셋째, 최적화된 방법의 처리 시간이 훨씬 짧다는 것이다. 또한 앞 장에서 제안했던 인접성 기법들이 특성을 분석할 의도로 실험이 진행 되었다. 앞 절에서 설명한 문제를 유분투 리눅스 상에서 듀얼 코어 1.86GHz의 CPU와 2G 메모리를 가지고 실험했다.

실험 데이터는 볼록면체(Convex) 형태의 데이터와 무작위(Random) 형태의 데이터, 그리고 인터넷에 공개된 일본 퍼즐 문제들을 가지고 실험했다. c10과 r10은 각각 10×10의 데이터를 의미한다. 표 2는 각 변환에 의해서 만들어진 CNF 식의 절 수이고, 표 3은 변수 수, 그리고 표 4는 만족가능성 처리기의 처리 시간이다. 표 2,3,4에서 SC^{M+} 는 순차 계수기를 이용해서 인접성을 나타낸 (10)의 방법이고, Opt는 4.1절에서 설명한 이진 트리를 이용한 이진 기수 조건을 최적한 (9)의 인접성을 적용한 방법이다.

볼록면체 형태의 데이터는 마름모 형태를 취하도록 했는데, 그 모양의 단순함 때문에 비교적 큰 예제도 처리할 수 있었다. 무작위 데이터의 경우는 비교적 많은 수의 절과 변수를 사용해서 논리식으로 변환되고 문제 자체의 복잡도 때문에 비교적 작은 예제를 처리하는데도 많은 시간이 소요되었다. 또한 일본 퍼즐의 경우는 볼록면체와 무작위 데이터의 중간에 위치하는 결과를 나타냈다. 그러나 일본퍼즐의 처리결과에 더 많은 의미를 부여할 수 있는 이유는 실제 의미 있는 형태의 이미지를 가지고 있기 때문이다.

먼저 순차 계수기에 인접성을 적용한 SA^{M+} 방식과 순차 계수기를 이용하되 $SC^{-1}+NR$ 과 $SC+PR$ 방식을 선택적으로 취하는 CS 방식을 비교하면 대부분의 경우 CS가 만들어내는 식의 절 수와 변수 수, 그리고 처리 속도가 월등하게 좋음을 알 수 있다. 그러나 r50과 100

표 2 변환된 CNF의 절 수

Problem	TA	SC^{M+}	CS	Opt
c10	3500	3576	1576	2076
c20	23740	30156	11340	14704
c30	73632	103736	36172	45968
c40	166000	248316	84136	103848
c50	312932	487896	161068	195660
c60	526404	846476	276360	328408
c80	1204400	2016636	645956	749136
c100	2296164	3950796	1251168	1423672
c120	3897648	6840956	2149744	2409016
r10	3537	2473	2211	1986
r15	12627	8613	8636	7405
r20	30065	19688	20024	17383
r25	66845	41932	43129	37690
r30	120641	72755	76027	67159
r35	200403	118452	124257	109991
r40	316406	181614	191192	171152
r45	480998	269496	283107	256506
r50	689632	377894	399634	362830
10×10	4251	3060	2616	2359
18×10	5343	6132	2269	3337
15×20	23399	18485	12613	12836
20×20	37453	26206	23159	21177
25×25	66835	52076	42844	40002
40×40	345895	209919	185750	177465
40×79	1030041	575821	503588	476048
50×100	1752110	1142766	916629	902986
100×100	5217099	2932747	2372118	2234458

×100의 예제에서는 SA^{M+} 방식이 더 좋은 결과를 보였다. 특히 100×100에서 처리 속도는 많은 차이를 보였는데, 그 원인은 CS 방식이 식 (7)과 (8)에서 설명한 일반적인 인접성을 사용하기 때문이고, 크기가 커질수록 좋은 결과를 낼 수 없게 된다.

다음으로 TA 방식과 Opt 방식을 비교하면, 역시 최적화된 Opt 방식이 역시 적은 절 수와 적은 변수로 논리식을 만들어 내며, 처리 시간 빨랐다. 특히 일본 퍼즐의 경우 평균적으로 생성된 논리식의 크기가 Opt 방식이 TA 방식의 50% 이하였고, 처리 속도 또한 절반에 가까웠다. 이는 Opt가 TA를 최적화 하되 인접성은 동일하게 표현할 수 있었기 때문에 크기가 큰 예제에서도 Opt가 TA보다 월등한 성능을 보일 수 있었다. 또한 순차 계수기를 이용한 방법보다 Opt의 방식이 더 효율적임을 알 수 있었다.

6. 결론 및 향후 연구

소프트웨어 공학의 여러 문제들을 명제 논리의 만족가능성 문제로 간주하는 여러 연구들은 최근 만족가능

표 3 변환된 CNF의 변수 수

Problem	TA	SA ^{M+}	CS	Opt
c10	780	1000	676	500
c20	3920	8000	4684	2760
c30	9780	27000	15044	7140
c40	18880	64000	34728	14160
c50	31100	125000	66784	23700
c60	46320	216000	114092	35640
c80	88320	512000	266768	69280
c100	144400	1000000	516920	114600
c120	214080	1728000	888324	171120
r10	861	633	759	474
r15	2777	1999	2702	1602
r20	6117	4243	5852	3463
r25	12449	8299	11549	6832
r30	21017	13683	19505	11527
r35	33263	20984	30174	17686
r40	49339	30528	44397	25971
r45	70503	43329	62797	36982
r50	97218	57107	84173	49396
10×10	994	763	858	545
18×10	1281	1721	844	813
15×20	4023	4284	3916	2331
20×20	6777	5733	6846	3909
25×25	11069	11677	13005	7026
40×40	42917	37369	45547	25445
40×79	90420	99630	114222	51469
50×100	175199	202626	203305	103707
100×100	343959	511821	530367	192117

표 4 만족가능성 처리기 처리 시간

Problem	TA	SA ^{M+}	CS	Opt
c10	0.003	0.003	0.000	0.004
c20	0.015	0.015	0.008	0.012
c30	0.045	0.059	0.021	0.024
c40	0.101	0.172	0.092	0.060
c50	0.195	0.404	0.096	0.108
c60	0.321	0.832	0.171	0.188
c80	0.715	5.115	0.420	0.404
c100	1.424	13.726	1.031	0.764
c120	2.441	31.787	1.733	1.320
r10	0.004	0.004	0.000	0.004
r15	0.016	0.012	0.012	0.004
r20	0.032	0.056	0.068	0.036
r25	0.140	0.176	0.172	0.136
r30	0.496	0.816	0.904	0.528
r35	10.668	2.188	4.300	1.836
r40	165.880	165.090	93.749	74.008
r45	40.234	25.985	11.244	38.674
r50	81.753	32.382	46.498	47.282
10×10	0.004	0.004	0.000	0.004
18×10	0.004	0.004	0.000	0.004
15×20	0.028	0.028	0.036	0.016
20×20	0.032	0.028	0.040	0.020
25×25	0.084	0.128	0.144	0.068
40×40	0.652	0.540	2.256	0.624
40×79	2.224	1.964	3.648	1.416
50×100	4.144	4.084	12.904	2.636
100×100	6.472	7.896	20.197	4.324

성 처리기들[5,6]의 높은 성능에서 비롯되었다. 실 세계의 문제들을 만족가능성 문제로 처리하려면 CNF로 변환되어야 하는데, 문제는 먼저 제약사항으로 기술된 후에 CNF로 변환되는데, 이진 기수 조건은 실 세계를 명제 논리의 만족가능성 문제로 변환하는데 자주 쓰이는 기법이다[9,10]. 본 논문은 이진 기수 조건과 같이 사용해서 문제를 CNF로 변환할 수 있는 인접성 조건을 제안했다.

또한 인접성을 기준에 제안된 이진 기수 조건들과 결합하는 방법들을 제안했고, 각 방법에서 최적화된 방법을 고안했다. 이렇게 고안된 방법의 효율성을 증명하기 위해서 사례 연구로써 이미지 복원 문제에 적용해 보았다. 실험은 세가지 형태의 데이터 즉, 블록면체와 무작위 데이터 그리고 일본 퍼즐을 대상으로 진행되었고 각각 생성된 식의 크기와 처리 시간을 비교해 보았다. 실험 결과 이진 트리를 이용한 방법과 순차 계수기를 이용한 방법 모두 최적화된 방식에서 좋은 성능을 보였고, 특히 이진 트리를 최적화한 방법이 가장 좋은 결과를 보였다. 이러한 연구는 앞으로 소프트웨어 모델 체킹에

도움을 줄 것으로 예상된다. 향후 제안한 방법들의 정당성을 증명하는 것과 다른 분야에 적용하는 문제들이 남아있다.

참고 문헌

- [1] S.M. Suhaib and D. BernerM, "Extreme Modeling in PROMELA: Formal Modeling and Verification of a Smart Building Control System," FERMAT Lab Technical Report No. 2003-11. Virginia Tech, Blacksburg, VA, December 2003.
- [2] M. Merritt, A. Orda, and S.R. Sachs, "Formal Verification of a Distributed Computer System," In Formal Methods in System Design Journal, Kluwer Academic Publisher, vol. 10, pp. 93-125, 1997.
- [3] G.J. Holzmann, The SPIN Model Checker, Addison Wesley, 2004.
- [4] E. M. Clarke, O. Grumberg and D. Peled, Model Checking, MIT Press, 1999.
- [5] M.W. Moskewicz, C. Madigan, Y. Zhao, L. Zhang and S. Malik, "Chaff: Engineering an Efficient SAT Solver," In Proceedings of Design Automation Conference, 2001.

- [6] <http://www.cs.chalmers.se/Cs/Research/Formal-Methods/MiniSat/>
- [7] A. Biere, A. Cimatti, E. Clarke, Ofer Strichman, and Y. Zhu, "Bounded Model Checking," Vol. 58 of Advances in Computers, 2003.
- [8] <http://alloy.mit.edu/>
- [9] O.Bailleux and Y. Boufkhad, "Efficient CNF encoding of Boolean cardinality constraints," in Proceedings of the CP 2003, vol. 2833, LNCS, 2003.
- [10] C. Sinz, "Towards an optimal CNF encoding of Boolean cardinality constraints," In Proceedings of the CP 2005, vol. 3709, LNCS, 2005.
- [11] B. J. Batenburg and W. A. Kusters, "A discrete tomography approach to Japanese puzzles," in Proceedings of the Belgian-Dutch Conf. Artificial Intelligence, 2004.
- [12] <http://en.wikipedia.org/wiki/Nonogram>
- [13] B. Benhamou, L. Sais and P. Siegel, "Two proof procedures for a cardinality based language in propositional calculus," in Proceedings of the STACS94, vol. 775, LNCS, 1994.
- [14] G. Kwon and W. Klieber, "Efficient CNF Encoding for Selecting 1 from N Objects," In the Proceeding of CFV07, 2007.

박 사 천

정보과학회논문지 : 소프트웨어 및 응용
제 35 권 제 9 호 참조

권 기 현

정보과학회논문지 : 소프트웨어 및 응용
제 35 권 제 2 호 참조