The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 23, No. 6, pp.125-132, Dec. 31, 2023. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2023.23.6.125
JIIBC 2023-6-19

최대-최대 빈도수 k-SAT 알고리즘

k-SAT Problem Algorithm Based on Maximum-Maximum Frequency

이상운*

Sang-Un Lee*

요 약 본 논문은 NP-완전으로 알려진 k-SAT 문제의 절의 수 m에 대해 $O(km)$의 다항시간 알고리즘을 제안하였다. 기존에 널리 알려진 DPLL은 문자 수 l에 대해 분기한정법의 전수탐색으로 해를 찾지 못하면 역추적을 수행하는 방식으로 최악의 경우 $O(2^l)$을 수행해야 한다. DPLL은 최소 빈도수 문자가 포함된 절을 참(T)으로 하도록 문자에 참(T) 또는 거짓(F)을 대입하여 해당 문자가 포함된 절을 제거하는 방식으로 SAT Solver의 근간을 이루고 있다. 제안된 알고리즘은 DPLL과는 반대로 부울 수식 f에 존재하는 최대 빈도수 문자 $_{\max}l$을 선택하고, $_{\max}(|l|,|\bar{l}|)=1$로 설정하고, $l\in c_i$ 절은 삭제하며, $\bar{l}\in c_i$절에서 $\bar{l}$를 삭제하는 방법을 적용하였다. 제안된 알고리즘을 다양한 k-SAT 문제들에 적용한 결과 기존의 DPLL 알고리즘보다 적은 횟수를 수행함을 알 수 있었다.

Abstract To NP-complete 3-SAT problem, this paper proposes a $O(nm)$ polynomial time algorithm, where n is the number of literals and m is the total frequency of all literals in equation f. Conventionally well-known DPLLs should perform $O(2^l)$ in the worst case by performing backtracking if they fail to find a solution in a brute-force search of a branch-and-bound for the number of literals l. DPLL forms the core of the SAT Solver by substituting true(T) or false(F) for a literal so that a clause containing the least frequency literal is true(T) and removing a clause containing that literal. Contrary to DPLL, the proposed algorithm selects a literal $_{\max}l$ with the maximum frequency and sets $_{\max}(|l|,|\bar{l}|)=1$. It then deletes $l\in c_i$ clause in addition to $\bar{l}$ from $\bar{l}\in c_i$ clause. Its test results on various k-SAT problems not only show that it performs less than existing DPLL algorithm, but prove its simplicity in satisfiability verification.

Key Words : Boolean satisfiability problem(SAT), Priority, Frequency, Max-Max frequency

Ⅰ. 서 론

충족 가능성 문제(Boolean satisfiability problem, SAT)는 주어진 부울 수식 f에 대해 문자에 참(true, T) 또는 거짓(false, F)의 진리 값을 할당하였을 경우 주어진 수식이 참으로 될 수 있는가를 결정하는 문제이다.[1,2] 문자들(literals, l) 변수들(variables)이 모여 하나의 절(clauses, c)을 이루며, 절들이 모여 부울 수식 f를 형성

*정회원, 강릉원주대학교 과학기술대학 멀티미디어공학과
접수일자 2023년 3월 3일, 수정완료 2023년 11월 9일
게재확정일자 2023년 12월 8일

Received: 3 March, 2023 / Revised: 9 November, 2023 / Accepted: 8 December, 2023
*Corresponding Author: sulee@gwnu.ac.kr
Dept. of Multimedia Eng., Gangneung-Wonju National University, Korea

한다. 각 문자 x는 양의 문자 x 또는 음의 문자 $\bar{x}$ (or $\neg x$)로 주어진다.

SAT 문제는 절 들 사이에 존재하는 연산자로 CNF (conjunctive normal form) 또는 DNF(disjunctive normal form)로 나뉜다. 절 사이에 논리곱(conjunction, AND, $\wedge$) 연산을, 절 내부의 문자들 간에는 논리합 (disjunction, OR, $\vee$) 연산을 하는 형태를 CNF라 하며, 역으로 절 들 간에는 논리합 연산을, 절 내부 문자들 간에는 논리곱 연산을 하면 DNF라 한다.

DNF-SAT 문제는 선형시간으로 해를 구할 수 있는 쉬운 문제에 속하는 반면에 CNF-SAT 문제는 다항시간으로 해를 구하는 알고리즘이 알려져 있지 않아 NP-완전으로 분류된 난제이다.

CNF-SAT 문제의 부울 함수 $f=T$이면 '충족 가능(satisfiable)'이라고 판별하며, $f=F$이면 '충족 불가능(unsatisfiable)'이라고 판별한다. 이와 같이 충족 가능성 여부를 판별하기 위해 문자들에 $[0|1]$ 또는 $[T|F]$의 진리 값을 할당하여 모든 절이 참이 되면 부울 수식 f는 참이 되며, 이 경우가 하나라도 존재하면 이를 '충족 가능'이라 한다. 왜냐하면 CNF는 절 내부는 OR 연산으로 어느 하나의 문자만 참이면 되지만 절들 간에는 AND 연산으로 모든 절이 참이 되어야만 한다. 반면에, 가능한 모든 진리 값을 부여하여도 수식 f가 참이 되지 못하면 '충족 불가능'이라 한다.

또한 SAT 문제는 하나의 절 내부에 존재하는 문자 수 k로 $k-$SAT 문제라 한다. 예로 $2-$SAT, $3-$SAT 등은 이를 의미한다.

f에 사용된 전체 문자 수를 n, 절의 개수를 m, 절에 포함된 문자수를 k라 하면 CNF-SAT인 $f=(p\vee\bar{q}\vee\bar{r})\wedge(p\vee q\vee s)$에는 p,q,r,s의 4개 문자가 사용되어 $n=4$이며, $(p\vee\bar{q}\vee\bar{r})$와 $(p\vee q\vee s)$의 2개 절이 있으므로 $m=2$이며, 하나의 절 내부에 문자는 최대 3개가 존재하여 $k=3$이다.[1,2]

SAT 문제는 명제논리, 데이터베이스 설계 및 분석, 자동화된 추론, 기계가 물체를 시각적으로 인식하는 Machine Vision, CAD, 로보틱스, 일정, 집적회로 설계 등의 분야에 적용되고 있다.[3]

최근까지 널리 사용되고 있는 SAT-Solver가 채택하고 있는 핵심 알고리즘은 1960년대 초반에 제안된 DPLL (Davis-Putnam-Logemann-Loveland)[4,5]이다.[6] DPLL 알고리즘은 2^n의 전-이진 트리를 형성하고, 트리의 좌측에는 $T(1)$, 우측에는 $F(0)$을 배정하는 형태로 최소 빈도수 문자부터 시작하여 $[0|1]$을 배정하면서 깊이우선 탐색(depth-first-search, DFS)을 수행한다. 탐색 과정에서 '충족불가능'이 발생하면 역추적(backtracking)으로 $[0|1]$을 $[1|0]$으로 변경하는 분기한정 법(branch-and-bound)의 전수 탐색 법(brute-force)의 일종이며, 최악의 경우 $O(2^n)$의 지수시간 수행 복잡도를 갖는다.

반면에 3-SAT 문제는 Cook[7]이 1971년에 NP-완전(NP-complete)으로 처음 제시하고, Karp[8,9]이 1972년에 21개 NP-완전 문제 중 첫 번째 문제로 제시한 이후 지금까지 정확한 해를 다항시간으로 얻지 못하고 있다. 2008년에 Gubin[10,11]은 $O(m^3)$의 다항시간 알고리즘을 제안하였다. Lee[12]는 3-SAT에 한정된 최소 빈도수 문자 우선 선택 방법을 제안하였다.

본 논문은 3-SAT에 한정하지 않고 일반적인 $k-$SAT 문제의 절의 수 m에 대해 수행 복잡도 $O(km)$으로 최대 빈도수 문자 $\max f(l)$를 탐색하여 "1"로 설정하며, 문자수 n에 대해 알고리즘을 $O(n)$ 수행하면서 한 번에 하나의 문자씩 삭제하는 수행 복잡도 $O(km)$인 알고리즘을 제안한다. 2장에서는 DPLL[5]의 $O(2^{(n+1)})$과 Gubin[10,11]의 $O(m^3)$ 알고리즘을 고찰해 본다. 3장에서는 $O(km)$의 최대-최대 빈도수(max-max frequency, MMF) 알고리즘을 제안한다. 4장에서는 다양한 사례들을 대상으로 제안된 알고리즘을 적용하여 본다.

II. 관련연구와 연구 배경

3장에서 제안되는 알고리즘의 적합성을 비교 분석하기 위해, 비교 대상으로 본 장에서는 식 (1)의 f_1 부울 수식[7]을 대상으로 DPLL[5]과 Gubin[10,11]의 $O(m^3)$ 알고리즘이 수행되는 과정의 복잡성을 소개한다.

$$f_1=(p\vee q\vee r)\wedge(\bar{p}\vee q\vee\bar{r})\wedge(p\vee\bar{q}\vee s)\wedge(\bar{p}\vee\bar{r}\vee\bar{s}) \quad (1)$$

f_1은 $n=4, c=4, k=3$이며, $pqrs=TTFF$(또는 1100)의 진리 값을 가질 경우 $f=T$로 '충족 가능'으로 판별된다.[7]

식 (1)의 정확한 해를 얻기 위해서는 2^n의 가능한 모든 경우 수에 대해 f의 진리 값을 결정하는 수행 복잡도 $O(2^n)$인 전수 탐색(exhaustive search)의 진리표를 작성해야 한다[6]. 이와 같이 n,m이 작은 값을 가질 경우 전

수조사 방법이 보다 효율적일 수 있다. 그러나 n이 큰 경우 NP-완전인 지수형태 알고리즘은 컴퓨터를 활용하여도 현실적으로 해를 구하기 어려워진다.

Gubin[10,11]은 f_1의 해를 구하기 위해 다음과 같이 절에 대한 호환성 행렬 법을 적용하였다.[12]

(1) 진리표 (truth-tables)에 기반하여 상삼각의 절 호환성 행렬 (clauses' compatibility matrices) $c_i : c_j$를 작성한다.

$c_1 : c_2$	$c_1 : c_3$	$c_1 : c_4$
	$c_2 : c_3$	$c_2 : c_4$
		$c_3 : c_4$

(2) 절 호환성 행렬 (clauses' compatibility matrices) $c_i : c_j, i \geq 2$에 대해 2번째와 3번째 행을 축소시킨다.

$c_1 : c_2$	$c_1 : c_3$	$c_1 : c_4$
	$c_1 \wedge c_2 : c_1 \wedge c$	$c_1 \wedge c_2 : c_1 \wedge c_4$
		$c_1 \wedge c_3 : c_1 \wedge c_4$

(3) 절 호환성 행렬 (clauses' compatibility matrices) $c_i : c_j, i \geq 2$에 대해 3번째 행을 축소시킨다.

$c_1 : c_2$	$c_1 : c_3$	$c_1 : c_4$
	$c_1 \wedge c_2 : c_1 \wedge c$	$c_1 \wedge c_2 : c_1 \wedge c_4$
		$c_1 \wedge c_2 \wedge c_3 : c_1 \wedge c_2 \wedge c_4$

(4) 어떤 하나 이상의 행렬에서"1"을 전혀 포함하고 있지 않으면 충족 불가능이며, 그렇지 않으면 충족 가능이다.

이 방법은 호환성 행렬을 작성하는 과정이 매우 난해하며, 매우 복잡한 행 축소 계산 과정을 거쳐야 한다. Gubin[10,11] 방법은 f_1에 대해 $O(m^3) = 4^3 = 64$회를 수행해야 한다. 또한 절의 개수 $m = {}_nC_3 \times 2^k$로 증가한다. 예로, 문자수 $n=3$인 경우 $m = 1 \times 8 = 8$, $n=4$인 경우 $m = 4 \times 8 = 32$, $n=5$인 경우 $m = 20 \times 8 = 160$으로 $n!/k! \times 2^k$로 증가한다. 따라서 절의 개수를 이용하는 $O(m^3)$의 알고리즘 수행 복잡도는 문자수 n이 증가하면 매우 복잡한 알고리즘이 된다.

DPLL 알고리즘[5,6]은 기본적으로 1개 문자만 포함하고 있는 절($|c_i| = 1$)인 단위 절(unit clause) 제거 규칙과 f에서 문자 x은 존재하지만 $\bar{x}$가 존재하지 않는 경우(이의 역도 성립한다.)인 순수 문자(pure literal) 제거 규칙을 적용한다. 따라서 단위 절이나 순수 문자가 존재할 경우, 이 문자를 $T(1)$로 설정하면 해당 절을 삭제할 수 있다. 만약, 단위 절이나 순수 문자가 존재하지 않으면 임의의 문자 x에 대해 $x=1, \bar{x}=0$을 할당하여 이 문자를 포함하고 있는 모든 절을 제거하여 부울 수식을 축소(단순화)시킨다. 만약, 부울 수식이 충족 불가능으로 판단되면 지금까지 선택된 문자의 진리 값을 반대로 적용하면서 충족 여부를 판단하는 역추적 법(backtracking)을 적용한다. 참고로 DPLL 알고리즘은 다음과 같이 수행된다.

(1) 부울 수식을 CNF (conjunctive normal form)로 변환시킨다.

(2) 단위 절이나 순수 문자가 존재하면 해당 문자 x (또는 $\bar{x}$)을 우선 선택하여 $x=1$ (또는 $\bar{x}=1$)로 설정하고 단순화 과정을 수행한다.

(3) 만약, 단위 절이나 순수 문자가 존재하지 않으면 임의의 문자 x를 선택한다. 만약, 선택된 문자가 x이면 $x=1, \bar{x}=0$로, $\bar{x}$이면 $x=0, \bar{x}=1$로 할당(True)하면 해당 절은 참으로 제거가 가능하여 단순화 과정을 수행한다.

(4) 부울 수식 f의 충족 불가능 여부를 판단한다.

만약, f= 충족 불가능이 결정되면 가장 최근에 할당된 문자의 진리값을 반대로 적용하여 단순화 과정을 다시 수행한다. 그래도 f=충족 불가능이면 현재까지 선택된 문자들을 역으로 추적하면서 진리값을 반대로 설정하면서 단순화 과정을 반복적으로 수행한다.

[단순화 과정]

(1) "1"값이 할당된 문자를 포함하고 있는 절 c_i를 삭제한다.

(2) "0"값이 할당된 문자를 포함하고 있는 절 c_i에서 해당 문자를 삭제한다.

표 1. f_1 문제의 DPLL 알고리즘
Table 1. DPLL Algorithm for f_1 Problem

수행횟수	문자 선택	부울 수식
$f_1 = (p \vee q \vee r) \wedge (\bar{p} \vee q \vee \bar{r}) \wedge (p \vee \bar{q} \vee s) \wedge (\bar{p} \vee \bar{r} \vee \bar{s})$		
1	p $p=1, \bar{p}=0$	$f_1 = (q \vee \bar{r}) \wedge (\bar{r} \vee \bar{s})$
2	q (순수문자) $q=1$	$f_1 = (\bar{r} \vee \bar{s})$
3	$\bar{r}$ (순수문자) $\bar{r}=1$	$f_1 = (\bar{s})$
4	$\bar{s}$ (순수문자) $\bar{s}=1$	$f_1 = \phi$
$f_1 = \phi$: 충족 가능		

f_1에 DPLL 알고리즘을 적용하여 보자. 여기서는 단위 절이나 순수 문자가 존재하면 우선 선택 법칙을 적용하고, 존재하지 않으면 첫 번째 절의 첫 번째에 존재하는 문자를 선택하는 규칙을 적용한다. f_1은 $k=3$인 3-SAT 문제로 단위 절은 존재하지 않는다. 또한 p,q,r,s의 4개 문자 모두 순수 문자는 존재하지 않는다. 따라서 표 1과 같이 첫 번째로 p를 선택하여 $p=1, \bar{p}=0$을 할당하면 p가 속한 절은 True로 삭제되고, $\bar{p}$가 속한 절에서는 $\bar{p}$만 삭제되어 $f_1=(q\vee\bar{r})\wedge(\bar{r}\vee\bar{s})$로 축소된다.

두 번째로 $f_1=(q\vee\bar{r})\wedge(\bar{r}\vee\bar{s})$ 에는 순수 문자 q가 존재한다. 따라서 순수 문자 q를 선택하여 $q=1$을 할당하면 $f_1=(\bar{r}\vee\bar{s})$로 축소된다. 세 번째로 $f_1=(\bar{r}\vee\bar{s})$에는 2개의 순수 문자 r,s가 존재하며, 여기서 $\bar{r}$를 선택하여 $\bar{r}=1$을 할당하면 $f_1=(\bar{s})$로 축소된다. 마지막으로 $\bar{s}=1$을 할당하면 $f_1=T$로 충족 가능이다. 결국, 4회가 수행된다.

f_1에 대한 DPLL은 탐색 과정에서 '충족 불가능'이 발생하지 않은 최적의 결과를 보였다. 반면에 탐색 과정에서 '충족 불가능'이 발생하여 역추적을 수행하는 사례로 Boulé[13]가 제시한 식 (2)의 충족 가능 문제에 대해 DPLL을 적용하여 본다.

$$f_2=(p\vee q\vee r)\wedge(p\vee q\vee\bar{r})\wedge(p\vee r\vee s)\wedge(\bar{p}\vee r\vee s)\wedge(\bar{p}\vee r\vee\bar{s})\wedge(\bar{q}\vee\bar{r}\vee\bar{s})\wedge(\bar{q}\vee\bar{r}\vee s) \quad (2)$$

f_2에 DPLL 알고리즘을 적용한 결과는 표 2에 제시되어 있다. 이 문제는 4회의 역추적 과정을 수행하여 총 11회 수행으로 충족가능으로 결정되었다.

만약, f에 단위 절이나 순수문자가 없다면, DPLL 알고리즘은 깊이우선 탐색(DFS)법의 일환인 분기한정 법으로 그림 1과 같이 $O(2^{(n+1)})$의 수행 복잡도를 갖는다.

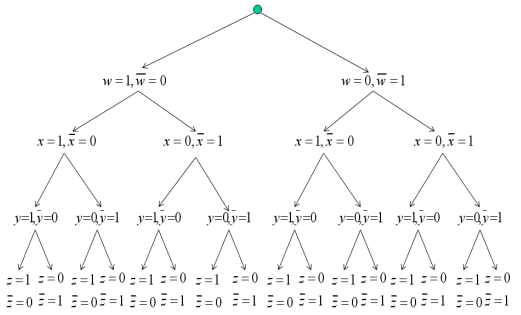

그림 1. 4-문자의 분기한정 법
Fig. 1. Branch-and-bound for 4 literals

표 2. f_2 문제의 DPLL 알고리즘
Table 2. DPLL Algorithm for f_2 Problem

수행횟수	문자 선택	부울 수식
1	p $p=1, \bar{p}=0$	$f_2=(q\vee\bar{r})\wedge(r\vee s)\wedge(r\vee\bar{s})\wedge(\bar{q}\vee\bar{r}\vee\bar{s})$ $\wedge(\bar{q}\vee\bar{r}\vee s)$
2	q $q=1, \bar{q}=0$	$f_2=(r\vee s)\wedge(r\vee\bar{s})\wedge(\bar{r}\vee\bar{s})\wedge(\bar{r}\vee s)$
3	r $r=1, \bar{r}=0$	$f_2=(\bar{s})\wedge(s)$: 충족 불가능
4	r $r=0, \bar{r}=1$	$f_2=(r\vee s)\wedge(r\vee\bar{s})\wedge(\bar{r}\vee\bar{s})\wedge(\bar{r}\vee s)$ $f_2=(s)\wedge(\bar{s})$: 충족 불가능
5	q $q=0, \bar{q}=1$	$f_2=(q\vee\bar{r})\wedge(r\vee s)\wedge(r\vee\bar{s})\wedge(\bar{q}\vee\bar{r}\vee\bar{s})$ $\wedge(\bar{q}\vee\bar{r}\vee s)$ $f_2=(\bar{r})\wedge(r\vee s)\wedge(r\vee\bar{s})$
6	$\bar{r}$(단위 절) $\bar{r}=1, r=0,$	$f_2=(s)\wedge(\bar{s})$: 충족 불가능
7	p $p=0, \bar{p}=1$	$f_2=(p\vee q\vee r)\wedge(p\vee q\vee\bar{r})\wedge(\bar{p}\vee q\vee\bar{r})\wedge$ $(p\vee r\vee s)\wedge(\bar{p}\vee r\vee s)\wedge(\bar{p}\vee r\vee\bar{s})\wedge$ $(\bar{q}\vee\bar{r}\vee\bar{s})\wedge(\bar{q}\vee\bar{r}\vee s)$ $f_2=(q\vee r)\wedge(q\vee\bar{r})\wedge(r\vee s)\wedge(\bar{q}\vee\bar{r}\vee\bar{s})\wedge$ $(\bar{q}\vee\bar{r}\vee s)$
8	q $q=1, \bar{q}=0$	$f_2=(r\vee s)\wedge(\bar{r}\vee\bar{s})\wedge(\bar{r}\vee s)$
9	r $r=1, \bar{r}=0$	$f_2=(\bar{s})\wedge(s)$: 충족 불가능
10	r $r=0, \bar{r}=1$	$f_2=(r\vee s)\wedge(\bar{r}\vee\bar{s})\wedge(\bar{r}\vee s)$ $f_2=(s)$
11	s (단위 절) $s=1$	$f_2=\phi$: 충족 가능
$f_2=\phi$: 충족 가능		

Ⅲ. 최대-최대 빈도수 알고리즘

본 장에서는 DPLL 알고리즘을 보다 간단히 수행하는 최대-최대 빈도수(MMF) 알고리즘을 제안한다. MMF를 제안하게 된 배경에는 주어진 부울 수식의 절의 개수를 가능한 빨리 제거하기 위해서는 DPLL의 최소 빈도 수 문자 우선 선택법의 역으로 최대 빈도 수 문자 우선 선택법이 보다 효율적이라는 아이디어에 기반하고 있다.

제안된 알고리즘은 부울함수 f에서 최대 빈도수 문자 $_{\max}l$을 선택하고, $_{\max}l$인 문자 l_j에 대해 다시 $_{\max}(|l_j|,|\bar{l}_j|)=1$로 설정하는 방법으로 DPLL 알고리즘에 비해 다음과 같은 차이점이 있다.

(1) DPLL 알고리즘은 단위 절과 순수 문자 우선선택 규칙은 적용하는데 반해, MMF 알고리즘은 단위 절 우선선택 규칙만 적용한다.

(2) DPLL 알고리즘은 단위 절이나 순수 무자가 없을 경우 임의의 문자를 선택하는데 반해, MMF 알고리즘은 단위 절이 없을 경우 최대 빈도수 문자 $_{\max}l$을 선택하는 방식을 채택한다. 단, $_{\max}l$에 대

해 $l=\bar{l}$인 경우 $l=1$로 배정하여 우선 수행하고, 충족 불가능으로 판별시 $\bar{l}=1$을 수행하여 충족 가능 여부를 재검증한다.

(3) DPLL 알고리즘은 최근까지 문자에 할당된 값을 순서대로 모두 갖고 있어야 분기한정법의 역추적 기능을 수행할 수 있다. 반면에, MMF 알고리즘은 단지 최대 빈도수 문자가 $l=\bar{l}$인 경우에만 저장하고 역추적 기능을 수행한다. 그렇지 않으면 역추적 기능을 수행하지 않는다.

제안된 MMF 알고리즘은 최대 빈도수 문자가 l만 유일하게 존재 시 문자수 n에 대해 $O(n)$으로 알고리즘이 수행된다. 만약, $l=\bar{l}$이고, $l=1$로 알고리즘 수행시 충족 불가능으로 판별되면 다시 $\bar{l}=1$을 수행하기 때문에 알고리즘 수행 복잡도는 $O(2n)$이 된다. 다만, 알고리즘 수행 과정에서 최대 문자수를 계속 확인하는데 $O(km)$이 소요된다. 따라서 $O(km)>O(2n)$이 되므로 제안된 알고리즘의 복잡도는 $O(km)$이다. MMF 알고리즘은 최대 빈도수 문자 l에 대해 $l=1$로 설정하여 해당 절을 삭제하기 때문에 1회 수행으로 하나의 문자가 축소되지만 절의 개수는 최대로 축소시킬 수 있는 장점이 있다. MMF 알고리즘은 그림 2와 같이 수행된다.

$S=\infty$. /* 충족 가능 여부 판단.
절 개수 : $c_i, i=1,2,\cdots,m$
문자 수 : $l_j, j=1,2,\cdots,n$
for $k=1$ to n
  if $\exists, |c_i|=1$ then $l\in c_i$ 문자에 대해 $l=1$, $\bar{l}=0$ 할당, $l\in c_i$ 절=1, $\bar{l}\in c_i$. /* 단위절 우선 선택
  else 각 문자 $l_j, j=1,2,\cdots,n$에 대해 $|l_j|$ 계산
    최대 빈도수 문자 $\max(|l_j|)$에 대해 $|l_j|$와 $|\bar{l}_j|$ 계산
    $\max(|l_j|,|\bar{l}_j|)=1, \min(|l_j|,|\bar{l}_j|)=0$로 설정
    만약, $l_j=\bar{l}_j$이면 $l_j=1$ 우선 설정, 만약, $l_j=1$로 설정, 충족 불가능 시 $\bar{l}_j=1$로 충족 가능 여부 재검증
    $l\in c_i$ 절 = 1, $\bar{l}\in c_i$는 $c_i=c_i\diagdown\bar{l}$
  end if
  if $\exists, c_i=0$ then $S=0$ /* 충족 불가능
  else if $\forall, c_i=0$ then $S=1$ /* 충족 가능
  else if $(l)\wedge(\bar{l})\in f$ then $S=0$ /* 충족 불가능
  else if $\exists, c_i\neq 0$ or 1 then loop
  endif
end

그림 2. MMF 알고리즘
Fig. 2. Max-Max frequency algorithm

MMF 알고리즘을 적용하여 f_1의 해를 구한 결과는 표 3에 제시되어 있다. f_1에서 최대 빈도수 문자는 p로 빈도수는 4이다. 또한, $|p|=2, |\bar{p}|=2$로 동일한 값을 갖는다. 따라서 p를 우선 수행하여 $p=1, \bar{p}=0$로 값이 할당되면 $f_1=(q\vee\bar{r})\wedge(\bar{r}\vee\bar{s})$로 축소된다. 다시 최대 빈도수 문자는 $r=2$, $|\bar{r}|=2$로 $\bar{r}=1$이 할당된다. 결국, $f_1=(1)\wedge(1)=1=1$로 $f_1=\phi$이 되어 충족 가능으로 판별된다. 따라서 최대 빈도수 문자 p에 대해 $|p|=2$, $|\bar{p}|=2$ 중 $\bar{p}=1, p=0$는 알고리즘이 재 수행되지 않아도 됨을 알 수 있다. 4개 변수($n=4$)인 f_1에 대해 DPLL 알고리즘은 4회 수행되는데 반해, MMF 알고리즘은 단지 2회 수행으로 충족 가능 여부를 판별하였다.

표 3. f_1 문제의 최대-최대 빈도수 알고리즘
Table 3. MMF Algorithm for f_1 Problem

수행횟수	문자 선택	부울 수식
$f_1=(p\vee q\vee r)\wedge(\bar{p}\vee q\vee\bar{r})\wedge(p\vee\bar{q}\vee s)\wedge(\bar{p}\vee\bar{r}\vee\bar{s})$		
$\max l_j: p=4, \lvert p\rvert=2, \lvert\bar{p}\rvert=2$		
1	$p=1, \bar{p}=0$	$f_1=(q\vee\bar{r})\wedge(\bar{r}\vee\bar{s})$
$\max l_j: r=2, \lvert\bar{r}\rvert=2$		
2	$\bar{r}=1, r=0$	$f_1=1$
f_1 : 충족 가능		

f_2에 MMF 알고리즘을 적용한 결과는 표 4에 제시되어 있다. DPLL 알고리즘은 11회 수행으로 충족 가능을 결정하는데 반해 MMF 알고리즘은 7회 만에 충족 가능 여부를 판별할 수 있음을 알 수 있다.

표 4. f_2 문제의 최대-최대 빈도수 알고리즘
Table 4. MMF Algorithm for f_2 Problem

수행횟수	문자 선택	부울 수식
$f_2=(p\vee q\vee r)\wedge(p\vee q\vee\bar{r})\wedge(\bar{p}\vee q\vee\bar{r})\wedge(p\vee r\vee s)\wedge(\bar{p}\vee r\vee s)\wedge(\bar{p}\vee r\vee\bar{s})\wedge(\bar{q}\vee\bar{r}\vee\bar{s})\wedge(\bar{q}\vee\bar{r}\vee s)$		
$\max l_j: r=8, \lvert r\rvert=4, \lvert\bar{r}\rvert=4$		
1	$r=1, \bar{r}=0$	$f_2=(p\vee q)\wedge(\bar{p}\vee q)\wedge(\bar{q}\vee\bar{s})\wedge(\bar{q}\vee s)$
$\max l_j: q=4, \lvert q\rvert=2, \lvert\bar{q}\rvert=2$		
2	$q=1, \bar{q}=0$	$f_2=(\bar{s})\wedge(s)$: 충족 불가능
3	$q=0, \bar{q}=1$	$f_2=(p)\wedge(\bar{p})$: 충족 불가능
4	$r=0, \bar{r}=1$	$f_2=(p\vee q)\wedge(p\vee s)\wedge(\bar{p}\vee s)\wedge(\bar{p}\vee\bar{s})$
$\max l_j: p=4, \lvert p\rvert=2, \lvert\bar{p}\rvert=2$		
5	$p=1, \bar{p}=0$	$f_2=(s)\wedge(\bar{s})$: 충족 불가능
6	$p=0, \bar{p}=1$	$f_2=(q)\wedge(s)$
7	q,s (단위 절) $q=1, s=1$	$f_2=1$: 충족 가능
f_2 : 충족 가능		

Ⅳ. 실험 및 결과 분석

본 장에서는 표 5에 제시된 다양한 문제들을 대상으로 제안된 MMF 알고리즘의 적용성을 평가해 본다. f_4, f_6과 f_{11}을 제외한 6개 수식은 모두 3-SAT이다. 표 5에 MMF 알고리즘을 적용한 결과는 표 6에 제시되어 있다.

표 5. 실험 사례
Table 5. Experimental Data

참고문헌	부울 수식	해
Gubin[11] Sutcliffe[14]	$f_3=(p\vee q\vee r)\wedge(p\vee q\vee\bar{r})\wedge(p\vee\bar{q}\vee r)\wedge(p\vee\bar{q}\vee\bar{r})\wedge(\bar{p}\vee q\vee r)\wedge(\bar{p}\vee q\vee\bar{r})\wedge(\bar{p}\vee\bar{q}\vee r)$	충족 가능
Gubin[11]	$f_4=\bar{p}\wedge\bar{q}\wedge\bar{r}\wedge(p\vee q)\wedge(p\vee r)\wedge(q\vee r)$	충족 불가능
Wikipedia[1]	$f_5=(x\vee y\vee\bar{z})\wedge(a\vee\bar{b}\vee c)\wedge(\bar{x}\vee b\vee z)$	충족 가능
Gubin[11]	$f_6=(p\vee q\vee r)\wedge(p\vee q\vee\bar{r})\wedge(\bar{p}\vee s)\wedge(\bar{p}\vee\bar{s})\wedge(\bar{q})$	충족 불가능
Jelliffe[15]	$f_7=(a\vee b\vee\bar{c})\wedge(\bar{b}\vee d\vee\bar{e})\wedge(\bar{a}\vee c\vee e)\wedge(\bar{c}\vee\bar{d}\vee\bar{e})$	충족 가능
Krasnoger [16]	$f_8=(p\vee q\vee r)\wedge(\bar{q}\vee\bar{r}\vee s)\wedge(\bar{p}\vee q\vee\bar{s})\wedge(p\vee\bar{q}\vee r)$	충족 가능
Lai[17]	$f_9=(p\vee\bar{q}\vee\bar{r})\wedge(\bar{p}\vee q\vee r)\wedge(p\vee q\vee r)$	충족 가능
Kochenberger [3]	$f_{10}=(p\vee q\vee r)\wedge(p\vee\bar{q}\vee r)\wedge(\bar{p}\vee q\vee\bar{r})\wedge(q\vee\bar{r}\vee s)\wedge(\bar{q}\vee r\vee s)(\bar{q}\vee\bar{r}\vee\bar{s})\wedge(q\vee s\vee t)\wedge(\bar{q}\vee r\vee t)\wedge(q\vee\bar{r}\vee t)\wedge(r\vee s\vee t)\wedge(r\vee\bar{s}\vee\bar{t})\wedge(\bar{r}\vee\bar{s}\vee\bar{t})$	충족 가능
Sutcliffe[14]	$f_{11}=(\bar{n}\vee\bar{t})\wedge(m\vee q\vee n)\wedge(l\vee\bar{m})\wedge(l\vee\bar{q})\wedge(\bar{l}\vee\bar{p})\wedge(r\vee p\vee n)\wedge(\bar{r}\vee\bar{l})\wedge(t)$	충족 불가능

표 6. 최대-최대 빈도수 알고리즘 적용 결과
Table 6. The Result of MMF Algorithm

$f_3=(p\vee q\vee r)\wedge(p\vee q\vee\bar{r})\wedge(p\vee\bar{q}\vee r)\wedge(p\vee\bar{q}\vee\bar{r})\wedge(\bar{p}\vee q\vee r)\wedge(\bar{p}\vee q\vee\bar{r})\wedge(\bar{p}\vee\bar{q}\vee r)$

수행횟수	문자 선택	부울 수식
	$\max l_j: p=q=r=7,\ \|p\|=4,\ \|\bar{p}\|=3$	
1	$p=1, \bar{p}=0$	$f_3=(q\vee r)\wedge(q\vee\bar{r})\wedge(\bar{q}\vee r)$
	$\max l_j: q=r=3,\ \|q\|=2,\ \|\bar{q}\|=1$	
2	$q=1, \bar{q}=0$	$f_3=(r)$
	r (단위 절)	
3	$r=1$	$f_3=1$
	f_3 : 충족 가능	

$f_4=\bar{p}\wedge\bar{q}\wedge\bar{r}\wedge(p\vee q)\wedge(p\vee r)\wedge(q\vee r)$

수행횟수	문자 선택	부울 수식
	$\bar{p},\bar{q},\bar{r}$ (단위 절)	
1	$p=0,\bar{p}=1$ $q=0,\bar{q}=1$ $r=0,\bar{r}=1$	$f_4=(0)\wedge(0)\wedge(0)$
	f_4 : 충족 불가능	

$f_5=(x\vee y\vee\bar{z})\wedge(a\vee\bar{b}\vee c)\wedge(\bar{x}\vee b\vee z)$

수행횟수	문자 선택	부울 수식
	$\max l_j: x=z=b=2,\ \|x\|=1,\ \|\bar{x}\|=1$	
1	$x=1, \bar{x}=0$	$f_5=(a\vee\bar{b}\vee c)\wedge(b\vee z)$
	$\max l_j: b=2,\ \|b\|=1,\ \|\bar{b}\|=1$	
2	$b=1, \bar{b}=0$	$f_5=(a\vee c)$
	$\max l_j: a=c=1,\ \|a\|=1,\ \|\bar{a}\|=0$	
3	$a=1,\ c=1$	$f_5=1$
	f_5 : 충족 가능	

$f_6=(p\vee q\vee r)\wedge(p\vee q\vee\bar{r})\wedge(\bar{p}\vee s)\wedge(\bar{p}\vee\bar{s})\wedge(\bar{q})$

수행횟수	문자 선택	부울 수식
	$\bar{q}$ (단위 절)	
1	$\bar{q}=1, q=0$	$f_6=(p\vee r)\wedge(p\vee\bar{r})\wedge(\bar{p}\vee s)\wedge(\bar{p}\vee\bar{s})$
	$\max l_j: p=4,\ \|p\|=2,\ \|\bar{p}\|=2$	
2	$p=1, \bar{p}=0$	$f_6=(s)\wedge(\bar{s})$: 충족 불가능
3	$p=0, \bar{p}=1$	$f_6=(r)\wedge(\bar{r})$: 충족 불가능
	f_6 : 충족 불가능	

$f_7=(a\vee b\vee\bar{c})\wedge(\bar{b}\vee d\vee\bar{e})\wedge(\bar{a}\vee c\vee e)\wedge(\bar{c}\vee\bar{d}\vee\bar{e})$

수행횟수	문자 선택	부울 수식
	$\max l_j: c=e=3,\ \|c\|=1,\ \|\bar{c}\|=2$	
1	$\bar{c}=1, c=0$	$f_7=(\bar{b}\vee d\vee\bar{e})\wedge(\bar{a}\vee e)$
	$\max l_j: e=2,\ \|e\|=1,\ \|\bar{e}\|=1$	
2	$e=1, \bar{e}=0$	$f_7=(\bar{b}\vee d)$
	$\max l_j: b=d=1,\ \|\bar{b}\|=1,\ \|d\|=1$	
3	$\bar{b}=1, d=1$	$f_7=1$
	f_7 : 충족 가능	

$f_8=(p\vee q\vee r)\wedge(\bar{q}\vee\bar{r}\vee s)\wedge(\bar{p}\vee q\vee\bar{s})\wedge(p\vee\bar{q}\vee r)$

수행횟수	문자 선택	부울 수식
	$\max l_j: q=4,\ \|q\|=2,\ \|\bar{q}\|=2$	
1	$q=1, \bar{q}=0$	$f_8=(\bar{r}\vee s)\wedge(p\vee r)$
	$\max l_j: r=2,\ \|r\|=1,\ \|\bar{r}\|=1$	
2	$r=1, \bar{r}=0$	$f_8=(s)$
	s (단위 절)	
3	$s=1$	$f_8=1$
	f_8 : 충족 가능	

$f_9=(p\vee\bar{q}\vee\bar{r})\wedge(\bar{p}\vee q\vee r)\wedge(p\vee q\vee r)$

수행횟수	문자 선택	부울 수식
	$\max l_j: p=q=r=3,\ \|p\|=2,\ \|\bar{p}\|=1$	
1	$p=1, \bar{p}=0$	$f_9=(q\vee r)$
	$\max l_j: q=r=1,\ \|q\|=1,\ \|\bar{q}\|=0$	
2	$q=1, \bar{q}=0$	$f_9=1$
	f_9 : 충족 가능	

$f_{10}=(p\vee q\vee r)\wedge(p\vee\bar{q}\vee r)\wedge(\bar{p}\vee q\vee\bar{r})\wedge(q\vee\bar{r}\vee s)\wedge(\bar{q}\vee r\vee s)\wedge(\bar{q}\vee\bar{r}\vee\bar{s})\wedge(q\vee s\vee t)\wedge(\bar{q}\vee r\vee t)\wedge(q\vee\bar{r}\vee t)\wedge(r\vee s\vee t)\wedge(r\vee\bar{s}\vee\bar{t})\wedge(\bar{r}\vee\bar{s}\vee\bar{t})$

수행횟수	문자 선택	부울 수식
	$\max l_j: r=11,\ \|r\|=6,\ \|\bar{r}\|=5$	
1	$r=1, \bar{r}=0$	$f_{10}=(\bar{p}\vee q)\wedge(q\vee s)\wedge(\bar{q}\vee\bar{s})\wedge(q\vee s\vee t)\wedge(q\vee t)\wedge(\bar{s}\vee\bar{t})$
	$\max l_j: q=5,\ \|q\|=4,\ \|\bar{q}\|=1$	
2	$q=1, \bar{q}=0$	$f_{10}=(\bar{s})\wedge(\bar{s}\vee\bar{t})$
	$\bar{s}$ (단위 절)	
3	$\bar{s}=1, s=0$	$f_{10}=1$
	f_{10} : 충족 가능	

$f_{11}=(\overline{n}\vee\overline{t})\wedge(m\vee q\vee n)\wedge(l\vee\overline{m})\wedge(l\vee\overline{q})\wedge(\overline{l}\vee\overline{p})\wedge(r\vee p\vee n)\wedge(\overline{r}\vee\overline{l})\wedge(t)$		
수행횟수	문자 선택	부울 수식
	t (단위 절)	
1	$t=1,\overline{t}=0$	$f_{11}=(\overline{n})\wedge(m\vee q\vee n)\wedge(l\vee\overline{m})\wedge(l\vee\overline{q})\wedge(\overline{l}\vee\overline{p})\wedge(r\vee p\vee n)\wedge(\overline{r}\vee\overline{l})$
	$\overline{n}$ (단위 절)	
2	$\overline{n}=1,n=0$	$f_{11}=(m\vee q)\wedge(l\vee\overline{m})\wedge(l\vee\overline{q})\wedge(\overline{l}\vee\overline{p})\wedge(r\vee p)\wedge(\overline{r}\vee\overline{l})$
	$_{\max}l_j : l=4, \lvert l\rvert=2, \lvert\overline{l}\rvert=2$	
3	$l=1,\overline{l}=0$	$f_{11}=(m\vee q)\wedge(\overline{p})\wedge(r\vee p)\wedge(\overline{r})$
	$\overline{p},\overline{r}$ (단위 절)	
4	$\overline{p}=1,p=0$ $r=1,r=0$	$f_{11}=(m\vee q)\wedge(0)$: 충족 불가능
5	$l=0,\overline{l}=1$	$f_{11}=(m\vee q)\wedge(l\vee\overline{m})\wedge(l\vee\overline{q})\wedge(\overline{l}\vee\overline{p})\wedge(r\vee p)\wedge(\overline{r}\vee\overline{l})$ $f_{11}=(m\vee q)\wedge(\overline{m})\wedge(\overline{q})\wedge(r\vee p)$
	$\overline{m},\overline{q}$ (단위 절)	
6	$\overline{m}=1,m=0$ $\overline{q}=1,q=0$	$f_{11}=(0)\wedge(r\vee p)$: 충족 불가능
	f_{11} : 충족 불가능	

본 논문에서 거론된 실험 데이터들에 대해 DPLL 알고리즘과 MMF 알고리즘의 수행횟수를 비교한 결과는 표 7에 제시되어 있다. 여기서 DPLL 알고리즘은 임의의 문자를 첫 번째 문자로 선택하는 방법을 적용한 결과이다.

표 7. DPLL과 MMF 알고리즘 수행 횟수
Table 7. Number of Trials for DPLL and MMF Algorithm

문제	해	수행횟수		증감
		DPLL	MMF	
f_1	SAT	4	**2**	-2
f_2	SAT	11	**7**	-4
f_3	SAT	3	**3**	0
f_4	UNSAT	1	**1**	0
f_5	SAT	2	3	+1
f_6	UNSAT	10	**3**	-7
f_7	SAT	4	**3**	-1
f_8	SAT	3	**3**	0
f_9	SAT	2	**2**	0
f_{10}	SAT	9	**3**	-6
f_{11}	UNSAT	37	**6**	-31

표로부터, MMF 알고리즘은 f_5를 제외하고는 나머지 10개 문제 중 6개 문제에서는 DPLL 알고리즘에 비해 적게 수행되었으며, 4개 문제에 대해서는 동일한 수행횟수를 보였다. f_5는 단위 문자가 포함된 경우로 DPLL 알고리즘이 보다 적은 수행횟수를 보였다. 또한, 특이한 점은 충족 불가능(UNSAT) 문제인 f_4를 제외하고, f_6과 f_{11}에 대해 DPLL 알고리즘은 가능한 모든 경우 수에 대해 역추적 기능을 수행하기 때문에 과다한 수행횟수가 발생하지만 제안된 MMF 알고리즘은 최대-최대 문자의 경우만을 고려하여 수행횟수를 크게 줄일 수 있었다. 여기서 f_4는 단위 절을 포함하고 있기 때문에 f_6과 f_{11}과 같은 현상이 발생하지 않았다.

결론적으로, $k-$SAT 문제에 대해 제안된 MMF 알고리즘은 DPLL 알고리즘에 비해 보다 적은 수행횟수로 충족 가능 여부를 판별할 수 있음을 알 수 있다.

V. 결론 및 향후 연구과제

본 논문은 k-SAT 문제의 충족 가능 여부를 절의 수 m에 대해 다항시간 $O(km)$으로 결정할 수 있는 알고리즘을 제안하였다.

제안된 알고리즘은 주어진 부울 함수 f에서 최대 빈도수 문자 $_{\max}l$을 선택하고, $_{\max}l$의 문자인 l_j에 대해 $_{\max}(\lvert l_j\rvert,\lvert\overline{l_j}\rvert)=1$로 설정하는 최대-최대 빈도수 선택 방법을 적용하였다.

제안된 알고리즘을 다양한 사례들에 적용 결과 DPLL 알고리즘에 비해 보다 적은 수행횟수로 충족 가능 여부를 결정할 수 있음을 보였다.

제안된 알고리즘은 $k-$SAT 문제에 대해 각 문자의 빈도수를 계산하는 과정을 절의 수 m에 대해 $O(km)$, 최대-최대 빈도수 문자를 포함한 절을 삭제하는 과정을 문자수 n에 대해 $O(n)$시간이 소요되어 알고리즘 수행 복잡도는 $O(km)$이다. 이는 Gubin의 $O(m^3)$과 DPLL 알고리즘의 $O(2^{(n+1)})$ 알고리즘에 비해 수행 시간을 단축시킬 뿐 아니라 간단하여 쉽게 적용할 수 있을 것이다.

$k-$SAT 문제에 대한 실험 데이터가 충분히 확보할 수 없는 관계로, 제안된 알고리즘은 크기가 소량인 데이터들에만 한정되었다. 추후 다량의 데이터에 대해 제안된 알고리즘의 적합성을 검증할 예정이다.

References

[1] Wikipedia, "Boolean Satisfiability Problem," http://en.wikipedia.org/wiki/Boolean_satisfiability_problem, Retrieved Jul. 2022.

[2] Wikipedia, "Satisfiability and Validity," http://en.wikipedia.org/wiki/Satisfiability_and_validity,

Retrieved Jul. 2022.

[3] G. Kochenberger, "Solution of Satisfiability Problems by Unconstrained Quadratic Programming," Hearin Center for Enterprise Science, http://hces.bus.olemiss.edu/conference/militaryworkshop-2003/Kochenberger.ppt, 2003.

[4] M. Davis and H. Putnam, "A Computing Procedure for Quantification Theory," Journal of the ACM, Vol. 7, No. 3, pp. 201-215, Jul. 1960, https://doi. org/10.1145/321033.321034

[5] M. Davis, G. Logemann, and D. Loveland, "A Machine Program for Theory Proving," Communications of the ACM, Vol. 5, No. 7, pp. 394-397, 1962, https://doi.org/10.1145/368273.368557

[6] M. Qasem, "SAT and MAX-SAT for the Lay- Researcher," School of Electronics and Computer Science, University of Sothampton, 2010.

[7] S. A. Cook, "The Complexity of Theorem Proving Procedures," Proceedings of the Third Annual ACM Symposium on the Theory of Computing, pp. 151-158, May 1971, https://doi.org/10.1145/800157. 805047

[8] R. M. Karp, "Reducibility Among Combinatorial Problems," in R. E. Miller and J. W. Thatcher (editors). Complexity of Computer Computations. pp. 85-103, New York: Plenum, 1972, https://doi. org/10.1007/978-1-4684-2001-2_9

[9] Wikipedia, "Karp's 21 NP-complete Problems, http://en.wikipedia.org/wiki/Karp's_21_NP-complete_problems, Retrieved Jul. 2022.

[10] S. Gubin, "A Polynomial Time Algorithm for 3-SAT," Cornell University Library, http://arxiv.org/ abs/cs.cc/0701023, 2008.

[11] S. Gubin, "Polynomial Time Algorithm for 3-SAT Examples of Use," Cornell University Library, http://arxiv.org/abs/cs.cc/0703098, 2008.

[12] S. U. Lee, "A 3-SAT Polynomial Time Algorithm Based on Minimum Frequency Literal-First Selection Method," The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 23, No. 1, pp. 157-162, Feb. 2023, https://doi.org/10.7236/JIIBC.2023.23.1.157

[13] M. Boulé, "MBSat Satisfiability Program and Heuristics Brief Overview," Electrical and Computer Engineering, McGill University, http://www.ece. mcgill.ca/~zzilic/649/MB.ppt, 2001.

[14] G. Sutcliffe, "CSC-648-Automated Theory Proving: The DPLL Algorithm," Department of Computer Science, University of Miami, 2010.

[15] R. Jelliffe, "3-SAT," http://lists.xml.org/archives/xml-dev/200103/msg00724.html, 2001.

[16] N. Krasnoger, "Computability (and Complexity): Reductions and Completeness," School of Computer Science, The University of Nottingham, 2009.

[17] K. J. Lai, "Design and Analysis of Algorithms," MIT, http://web.mit.edu/k_lai/www/6.046/r12-handout.pdf, 2008.

저 자 소 개

이 상 운(정회원)

- 1987년 : 한국항공대학교 항공전자공학과 (학사)
- 1997년 : 경상대학교 컴퓨터과학과 (석사)
- 2001년 : 경상대학교 컴퓨터과학과 (박사)
- 2003년 : 강원도립대학 컴퓨터응용과 전임강사
- 2004년 ~ 2007.2 : 국립 원주대학 여성교양과 조교수
- 2007.3 ~ 2015.3 : 강릉원주대학교 멀티미디어공학과 부교수
- 2015.4 ~ 현재 : 강릉원주대학교 멀티미디어공학과 정교수
- 관심분야 : 소프트웨어 프로젝트 관리, 개발 방법론, 분석과 설계 방법론, 시험 및 품질보증, 소프트웨어 신뢰성, 인공지능과 빅데이터분석, 최적화 알고리즘
- e-mail : sulee@gwnu.ac.kr