DNA 컴퓨팅을 이용한 삼단논증의 결론 증명

박의준[0] 이인희 장병탁
서울대학교 대학원 인지과학 협동과정[0]
서울대학교 컴퓨터 공학부 바이오지능 연구실
theory94@snu.ac.kr[0], {ihlee, btzhang}@bi.snu.ac.kr

Proof of Conclusion in Syllogism with DNA Computing

Eui-Jun Park[0] In-Hee Lee Byoung-Tak Zhang
Interdisciplinary Program in Cognitive Science[0],
Biointelligence Laboratory, School of Computer Science and Engineering
Seoul National University, Seoul 151-742, Korea

요 약

본 논문에서는 논리학에서 전통적으로 다루어 온 패턴인 삼단논증의 결론을 DNA 컴퓨팅을 이용해 증명해 내는 방법을 제시한다: 연역 장치로 진리나무 방법의 하나(resolution refutation)를 사용하기 위해서, 삼단논증의 전제들과 결론의 부정을 예화시킨 후 CNF 형태로 바꾸어 준다. 그리고 이것을 이중 가닥의 DNA 분자로 디자인한 후, 해소 반응을 통해 모순, 즉 닐(nil)을 발견하게 되면, 증명은 완료된다.

1. 서 론

인간의 논리적인 추론의 모델링이라는 점에서 정리 증명기(theorem prover)는 고전적 AI의 대표적인 연구 대상 중 하나고, 넓은 응용 가능성 또한 갖고 있다. 그러나 다루고자 하는 논리식들의 복잡도가 커짐에 따라 정리 증명을 수행하는 데 걸리는 시간은 지수 함수적으로 증가하게 되므로, 병렬 처리가 이루어지는 정리 증명기들이 제안되었고, 나아가 최근에는 DNA 컴퓨팅을 이용한 정리 증명기에 관한 연구 결과들도 발표되었다[1,2].

한편 DNA 컴퓨팅을 이용하여 술어 논리의 정리들을 증명하고자 하는 시도도 또한 있어 왔다. 여기에는 Prolog 구현[3], 혼 절(Horn clause) 계산[4, 5] 등이 포함된다. 그러나 이들 모두는 완벽하다고 보기 힘들다. [3]의 문제점은 일정 수 이상의 변수가 포함된 식, 그리고 어느 정도 이상 복잡한 식은 계산할 수 없다는 것이다. [4]의 단점은 원치 않는 에러가 발생할 가능성이 높다는 것이며, [5]은 실험 과정 상의 어려움 때문에 실제로 구현되지 못한 상태이다.

이렇든 술어 논리 증명기의 구현은 쉽지 않은데, 그것은 일차적으로는 어떤 술어 논리 체계의 일반적인 논리식들과 연역 장치(deductive apparatus)를 DNA 컴퓨팅이 가능하도록 표현하는 일이 명제 논리의 경우와는 달리 만만치 않기 때문으로 보인다. 따라서 우리는 이 글에서 작업의 범위를 좁혀, 일차 술어 논리의 특수한 경우로서 아리스토텔레스의 삼단논증을 다루고자 한다.

2. DNA 컴퓨팅을 이용한 명제 논리 정리 증명

명제 논리에 대한 정리 증명기는 이미 구현된 바 있다[2]. [2]에서는 증명을 위해, 주어진 논리식들이 연언표준형(conjunctive normal form; CNF)의 형태를 갖는다고 가정하였다. 그리고 주어진 논리식들 중 문장 문자(sentential letter)를 하나만 포함한 것은 헤어핀(hairpin) 구조의 DNA 분자로, 여러 개의 문장 문자를 포함한 것은 포함하고 있는 문장 문자의 개수만큼의 가지(branch)를 가진 구조의 DNA 분자로 표현하였다. 이 때, 표현하고자 하는 문장 문자들 자체는 염기 서열(sequence)의 단일 가닥으로 가지나 헤어핀 구조의 끝에 위치하게 된다. 그리고 증명하고자 하는 결론의 부정(negation)은 단일 가닥의 염기 서열로 나타내었다.

서로 다른 부호를 가진 문장 문자는 상보적인 서열로 나타내었기 때문에 문장 문자를 해소하는(resolve) 과정은 상보 결합으로 나타낼 수 있다. 이 경우 닐(nil)은 당연히 모든 부분이 이중 가닥인 DNA 분자로 표현 될 것이다. 그리고 논리식들을 표현한 DNA의 구조상, hybridization 결과 단일 가닥인 부분을 하나도 포함하지 않게 된 분자는 하나의 긴 가닥을 이루고 처음과 끝은

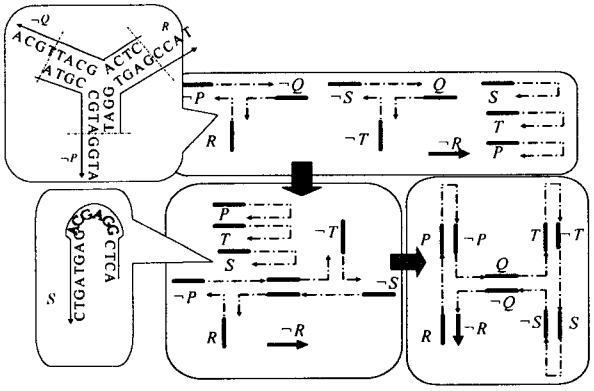

그림 1. 문제 "P∧Q→R, S∧T→Q, S, T, P, R?"에 대한 resolution refutation 과정의 모식도
(Preliminary Proc. of DNA8에서 발췌)

증명하기를 원했던 논리식과 그것의 부정이 될 것이다 (그림 1 참조). 이러한 DNA 가닥은 hybridization, ligation, PCR, gel electrophoresis를 차례로 거쳐 발견되었다. 만약 실험 중 해소(resolution)가 이루어지지 않은 분자가 있다면 ligation을 거친 후에도 가닥 중간에 이어지지 않은 부분이 있어서 PCR을 통해 제대로 증폭되지 않을 것이므로, 닐(nil)이 발생했다면 PCR 단계에서 이에 해당되는 분자들만이 증폭될 것이다. 일단 증폭이 이루어진 다음에는 gel electrophoresis를 통해 닐(nil)이 존재하는 지의 여부와, 닐(nil)이 유도된 과정에 대한 정보를 얻을 수 있게 된다.

3. DNA 컴퓨팅을 이용한 삼단논증의 결론 증명

3.1. 아리스토텔레스의 삼단논증

아리스토텔레스의 삼단논증에서는 주어진 두 개의 전제들로부터 하나의 결론을 이끌어낸다. 이 때, 전제들, 그리고 결론은 반드시 다음과 같은 형식의 문장 중 하나이다:

A(보편 긍정): 모든 a는 b이다.
I (특수 긍정): 어떤 a는 b이다.
E(보편 부정): 어떤 a도 b가 아니다.
O(특수 부정): 어떤 a는 b가 아니다.

A, I, E, O 문장들의 다양한 조합이 삼단논증의 패턴을 결정하게 된다. 이들 패턴들 중 가장 기본적인 것은 4 가지이며, 각각 'Barbara', 'Celarent', 'Darii', 'Ferio'라 불린다:

(1) Barbara
A: 모든 새는 동물이다.
A: 모든 펭귄은 새이다.
A: 그러므로 모든 펭귄은 동물이다.
(2) Celarent
E: 어떤 새도 식물이 아니다.
A: 모든 펭귄은 새이다.
E: 그러므로 어떤 펭귄도 식물이 아니다.
(3) Darii
A: 모든 새는 동물이다.
I: 어떤 날아다니는 것은 새이다.
I: 그러므로 어떤 날아다니는 것은 동물이다.
(4) Ferio
E: 어떤 박쥐도 새가 아니다.
I: 어떤 날아다니는 것은 박쥐이다.
O: 그러므로 어떤 날아다니는 것은 새가 아니다.

3.2. 논리식의 표현 방법

삼단논증은 많아야 세 개념을 지닌다. 결론의 술어인 대개념(major term), 결론의 주어인 소개념(minor term), 그리고 두 전제에 공통되는 매개념(middle term)이 그것이다. 그러나 현대 논리학의 관점에서 볼 때, 이들은 단지 세 개의 술어(predicate)에 불과하다. 곧 A, I, E, O 문장들은 다음과 같이 다시 쓰여질 수 있다:

A: $(\forall x)(Px \supset Qx)$
I : $(\exists x)(Px \wedge Qx)$
E: $(\forall x)(Px \supset -Qx)$
O: $(\exists x)(Px \wedge -Qx)$

우리는 [2]에서와 동일한 방법, 곧 resolution refutation을 써서 증명을 진행시키고자 한다. 그러므로 이들 논리식들 및 각각에 대한 부정들만 DNA 분자로 표현해낼 수 있으면 된다. 핵심적 아이디어는 매우 작은 수의 대상들만으로 논의 영역(domain)을 한정하는 데 있다. 예컨대 A의 경우 논의 영역에 a와 b의 두 대상만이 있다고 가정하여,

$$(\forall x)(Px \supset Qx) \equiv (Pa \supset Qa) \wedge (Pb \supset Qb)$$

와 같이 원래의 논리식을 예화(specification)시킨다. 보편 양화사와 존재 양화사에 대응하는 논리적 연결사는 각각 '$\wedge$'와 '$\vee$'이다. 따라서 DNA 컴퓨팅을 이용하여 Ferio를 보이기 위해서는 전제들과 결론을 일단 다음과 같이 표현한 후, 다시 CNF 형태로 바꾸어 주어야 한다:

$(\forall x)(Px \supset -Qx) \equiv (Pa \supset -Qa) \wedge (Pb \supset -Qb)$
$(\exists x)(Rx \wedge Px) \equiv (Ra \wedge Pa) \vee (Rb \wedge Pb)$
$-(\exists x)(Rx \wedge -Qx) \equiv (-Ra \vee Qa) \wedge (-Rb \vee Qb)$

여기서 문제는 이런 식으로 전제들을 표현하는 것이 과연 정당한가 하는 점이다. 그러나 우리는 적어도 삼단논증의 경우엔 위와 같은 표현 방법이 정당하다고 본다. 그것은 술어 논리의 연역 장치를 구성하는 두 개의 규칙, US(보편 예화, Universal Specification)와 ES(존재 예화, Existential Specification)가 갖는 근본적인 의미를 생각한다면 쉽게 납득할 수 있다. US 규칙을 적용한 결과, 예컨대 'Fa'는 어떤 특정한 개체 a에 관한 기술이 아니다. 의미론적으로 볼 때, 'Fa'는 논의 영역을 어떻게 잡고 거기서 어떤 개체를 선택한다 해도, 그 임의의 개체는 F라는 속성을 갖고 있다는 내용을 함축한다. 이 때 임의의 개체를 나타내는 이름이 다름 아닌 'a'인 것이다. 또 ES 규칙을 적용한 결과, 예컨대 'Pb'는 어떤 특정한 개체 b에 관한 기술이 아니다. 이것의 의미는 논의 영역에 P라는 속성을 갖고 있는 대상이 분명히 있어서 그것에 임시로 'b'라는 이름을 붙이겠다는 것이다. 그러므로 보편 예화의 결과로 얻게 되는 이름과 존재 예화의 결과로 얻게 되는 이름이 반드시 동일하지는 않을 정도로만 논의 영역을 충분하게 잡아 준다면, 우리의 방법에는 아무런 문제가 없다고 믿는다. 이 문제의 경우 논의 영역이 {a}이면 곤란하지만, 그밖에 {a, b}, {a, b, c}, {a, b, c, d}, … 는 모두 가능하다.

4. DNA를 이용한 구현

예화와 CNF로의 변형이 끝난 후의 논리식들은 [2,

5]에서 문장 문자 두 개의 논리합으로 구성된 절(clause)을 나타내는 것과 유사한 형태로 표현될 수 있다. 이것은 A, I, E, O 문장들 및 그 각각의 부정을 예화시킨 후 CNF 형태로 변형해 보면 쉽게 알 수 있다. 각각의 리터럴(literal, 원자식(atomic formula) 혹은 그것의 부정)들은 마치 명제 논리에서의 문장 문자의 경우와 같이 나타내면 된다. 이제 어떤 원자식, 예컨대 'Qa'를 특정한 염기 서열 α로 표현했을 때, 그것의 부정 '-Qa'는 α와 상보적인 염기 서열로, 그리고 'Qa'와 무관한 기타의 리터럴들은 역시 α와 무관한 염거서열로 디자인하기로 하겠다. 따라서 CNF 형태로 변형했을 때의 각각의 절들은 그림 2와 같이 표현된다. 그림 2는 '-Ra ∨ Qa', '-Qa ∨ Pa'에 대응하는 염기 서열을 보여준다.

그림 2. Clause의 표현

그 결과 두 절들 사이의 해소(resolution)는 [2,5]에서와 마찬가지로 두 절들을 나타내는 DNA 분자의 상보결합으로 이루어지게 된다. 따라서 만약 최종적으로 닐(nil)이 발생하면 닐은 리터럴을 포함하지 않는 절이므로, 리터럴에 해당하는 단일 가닥 부분이 없는, 이중 가닥의 고리 형태의 DNA 분자가 생성될 것이다. 이 고리 형태의 DNA 분자는 exonuclease라는 효소를 사용해도 분해되지 않는다. 따라서 해소 반응이 종료된 상태에서 용액에 exonuclease를 첨가한 후 분해되지 않고 남아있는 DNA 분자가 있는지를 조사하면, 닐이 발생하였는지 여부를 알 수 있게 된다. 이와 같은 방법을 통해, 삼단 논증의 결론이 두 전제로부터 도출될 수 있는 것인지를 판단할 수 있게 된다.

5. 결 론

지금까지 아리스토텔레스의 삼단논증의 결론을 DNA 컴퓨팅을 이용해 증명해 내는 방법을 제시하였다. 이것의 구현은 현재의 기술 수준으로도 충분히 가능하리라 생각한다. 삼단논증은 논리학에서 전통적으로 다뤄 온 패턴이기도 하지만, 일상생활에서도 흔히 많이 사용되고, 또 객체지향 시스템에서 상속(inheritance) 규칙의 근간이 되는 것이기도 하기 때문에 응용의 측면에서도 또한 의미를 가질 수 있다고 생각한다. 이런 점에서 우리의 증명기가 갖는 의의를 발견할 수 있을 것으로 보인다.

참고 문헌

[1] Wasiewicz, P., Janczak, T., Mulawka, J.J. and Pluciennicak, A., *The Inference Based on Molecular Computing*, International Journal of Cybernetics and Systems, 31, 3, pp. 283-315, 2000.

[2] Lee, I.-H., Park, J.-Y., Jang, H.-M., Chai, Y.-G. and Zhang., B.-T., *DNA Implementation of Theorem Proving with Resolution Refutation in Propositional Logic*, Preliminary Proceedings of Eighth International Meeting on DNA Based Computers, pp. 251-260.

[3] Mihalache, V., *Prolog Approach to DNA Computing*, Proceedings of the International Conference on Evolutionary Computation, pp. 249-254, 1997.

[4] Kobayashi, S., *Horn Clause Computation with DNA Molecules*, Journal of Combinatorial Optimization, 3, pp. 277-299, 1999.

[5] Uejima, H., Hagiya, M. and Kobayashi, S., *Horn Clause Computation by Self-Assembly of DNA Molecules*, Preliminary Proceedings of the Seventh International Meeting on DNA Based Computers, pp. 63-71, 2001.

[6] Jeffrey, R., *Formal Logic: Its Scope and Limits*, 2nd ed., McGraw-Hill Book Company, 1981.

[7] Mates, B., *Elementary Logic*, 2nd ed., Oxford University Press, 1972.