

데이터 의존적 AI를 넘어: 인지 아키텍처 기반 기호적 문제 해결 시스템의 설계

이석기⁰, 아자마트 메데트베크프, 최경민, 로라 마스카렐, 김선동
광주과학기술원 (GIST)

{sklee1103, azamatmd, youvegotmail}@gm.gist.ac.kr, {lmascarell, sundong}@gist.ac.kr

Beyond Data-Dependent AI: Design of a Symbolic Problem-Solving System Based on Cognitive Architecture

Seokki Lee⁰, Azamat Medetbekov, Kyungmin Choi, Laura Mascarell, Sundong Kim
Gwangju Institute of Science and Technology (GIST)

{sklee1103, azamatmd, youvegotmail}@gm.gist.ac.kr, {lmascarell, sundong}@gist.ac.kr

요약

François Chollet의 지능 정의에 따르면 높은 지능은 적은 사전 지식과 경험으로 높은 일반화 난이도의 과제를 해결하는 효율에 있다. 그러나 현대 대규모 언어 모델(LLM) 기반 시스템은 방대한 데이터와 연산 자원에 의존하여 이 정의에서의 분모를 지속적으로 키우는 방향으로 발전해왔다. 본 논문은 이와 대비되는 방향으로, 높은 지능을 달성하기 위한 세 가지 설계 조건을 제시하고, 이를 구현한 기호적 문제 해결 시스템 ARBOR(Abstraction Reasoning with Bottom-up Organized Rules)를 제안한다. ARBOR는 SOAR 인지 아키텍처의 문제 해결 구조, 계층적 지식 그래프 ARCKG(ARC Knowledge Graph), 그리고 반-통합(anti-unification) 기반 지식 일반화를 결합하여 신경망 없이 동작한다. 초기 실험에서 ARBOR는 82개의 규칙으로 72개의 문제를 해결하였으며, 규칙 커버리지(규칙 하나가 커버하는 문제 수) 0.88을 달성하였다. 이 결과는 규칙 기반 접근의 가능성을 보이는 동시에, 커버리지를 1 이상으로 높이기 위한 내부 지식 일반화 메커니즘의 필요성을 시사한다.

1. 서론*

인공지능의 일반적 추론 능력을 평가하기 위해 Chollet [1]이 제안한 ARC(Abstraction and Reasoning Corpus)는 소수의 입출력 예시로부터 변환 규칙을 귀납하여 처음 보는 테스트 입력에 적용하는 벤치마크이다. Chollet은 지능을 과제의 일반화 난이도를 시스템이 사용한 사전 지식과 경험으로 나눈 비율, 즉 기술 습득 효율로 정의한다 [1]. 동일 과제를 해결하더라도 더 적은 사전 지식과 경험으로 해결할수록 높은 지능을 보인 것으로 간주된다.

최근 ARC의 후속 벤치마크인 ARC-AGI-2에서 일부 대형 언어 모델이 공개 평가셋 기준 80% 안팎의 성능을 보고한 반면, 공식 대회 조건 하 최고 성적은 30%에 못 미친다 [2]. 이 격차는 현재 고성능 AI의 성능이 데이터와 연산 규모에 강하게 의존하는 구조적 특성을 드러낸다. Chollet의 정의에 따르면 풍부한 사전 지식과 대규모 학습 경험은 지능 측정의 분모에 해당하며, 충분한 자원만 주어진다면 임

의 수준의 성능을 달성할 수 있지만 이는 시스템 고유의 일반화 능력을 반영하지 않는다. 본 논문은 이러한 관점에서 높은 지능을 달성하기 위한 세 가지 설계 조건을 도출하고, 이를 구현한 기호적 문제 해결 시스템 ARBOR를 제안한다. ARC는 적은 예시로부터 규칙을 귀납해야 한다는 특성상 이 조건들을 검증하기에 적합한 테스트베드이다.

2. 설계 조건과 관련 연구

높은 기술 습득 효율은 더 적은 사전 지식과 경험으로 더 높은 일반화 난이도의 과제를 해결하는 것을 의미한다. 이를 달성하기 위해 시스템은 새로운 문제를 효과적으로 풀 수 있어야 하고, 지식을 재사용 가능한 형태로 저장해야 하며, 기존 지식을 추상화하여 분모의 증가를 억제해야 한다. 이 세 요건으로부터 다음의 설계 조건이 도출된다.

조건 1: 막히는 지점을 진단하고 돌파할 수 있어야 한다. 새로운 문제를 풀다 막혔을 때 무엇이 부족한지를 진단하고 그 지식을 채우는 방향으로 탐색을 이어가야 한다. ARBOR는 이를 위해 SOAR [3]의 설계 원리를 채택한다. SOAR는 적절한 연산자를 선택할 수 없을 때 충돌 상태를 발생시키고, 이를 하위 목표 생성으로 연결하여 부족한 지

* 본 연구는 IITP 디지털혁신기술 국제공동연구(RS-2024-00445087), 과학기술원 InnoCORE 사업(26-InnoCORE-01)에 의해 수행되었습니다.

식을 끌어오는 규칙 기반 결정 사이클을 갖는다. 각 풀이 단계가 명시적 규칙의 적용으로 이루어지므로 추론 과정의 설명 가능성이 확보되며, 통계적 패턴 매칭 없이도 더 적은 사전 경험으로 더 많은 문제를 해결하는 데 기여한다.

조건 2: 지식이 명시적이고 재사용 가능한 형태로 표현되어야 한다. LLM은 지식을 고차원 벡터 공간에 압축하여 저장하는데, 이는 대규모 데이터에서 패턴을 포착하는 데 유리하지만 연산 후 원래 의미를 해석하거나 특정 지식을 직접 조작하기 어렵다. 반면 기호적 표현은 지식을 명시적 구조로 저장하여 설명 가능하고 직접 조작 가능하다. ARBOR는 계층적 지식 그래프 ARCKG를 의미 메모리로 채택하여 ARC 문제를 5계층 노드 구조로 분해하고, 입출력 간 관계를 공통점(COMM)과 차이점(DIFF)으로 근사 없이 명시적으로 기록한다. ARG [4]는 그리드를 객체 그래프로 추상화하는 방향에서 이 조건에 근접하지만 사전 정의된 그래프 구조에 의존하며, ARCKG [5]는 계층적 지식 그래프 표현의 초기 시도로 ARBOR는 이를 5계층으로 확장하고 관계 차수 체계를 새롭게 정립한다. 이를 통해 동일한 지식으로 더 많은 문제를 커버할 수 있어 분모 증가 없이 기술 습득 효율을 높이는 기반이 된다.

조건 3: 지식이 내부에서 스스로 추상화되어야 한다. 규칙이 쌓여도 통합되지 않으면 새로운 문제마다 새 규칙이 추가되어 분모가 계속 커진다. DreamCoder [6]는 wake-sleep 방식으로 DSL 라이브러리를 점진적으로 확장하며 이 방향에 가장 근접하지만, 신경망 기반 인식 모델에 의존하여 데이터 의존성이 잔존한다. ARBOR는 반-통합 [7]을 이 조건의 구현 메커니즘으로 채택한다. 반-통합은 단 두 규칙의 공통 골격만으로 더 일반적인 규칙을 도출하는 형식적 연산으로, 대규모 데이터 없이 순수 기호적으로 동작한다. 이를 통해 기존 규칙을 추상화하여 더 넓은 범위를 커버하는 규칙으로 통합할 수 있으므로, 분모를 줄이면서 기술 습득 효율을 높일 수 있다.

3. ARBOR 아키텍처

3.1 전체 구성 및 작동 흐름

ARBOR는 2절에서 제시한 세 조건을 구현하는 시스템으로, 그림 1에 전체 구조가 나타나 있다. 그림 1의 좌측은 SOAR [3]에서 기반한 메모리 구조로, 작업 메모리를 중심으로 의미 메모리, 절차 메모리, 일화 메모리가 연결된 구조를 나타내며 각 메모리의 역할은 3.2절에서 설명한다. 그림 1의 우측은 의미 메모리의 구체적 구현체인 5계층 지식 그래프 ARCKG의 구조를 나타내며, 3.3절에서 상세히 설명한다. 3.4절에서는 이 구성 요소들이 상호작용하며 문제를 해결하는 전체 흐름을 설명한다.

3.2 메모리 구조와 역할

작업 메모리(Working Memory)는 ARBOR 추론의 중추적 요소로, 현재 풀이 상태를 실시간으로 유지하는 단기 메모리의 성질을 갖는다. 문제가 입력되면 해당 정보가 작업 메모리에 적재되고, 이후 모든 추론 프로세스는 작업 메모리의 상태를 읽고 갱신하는 방식으로 진행된다. 작업 메모리의 내용은 〈식별자, 속성, 값〉 삼중쌍의 집합으로

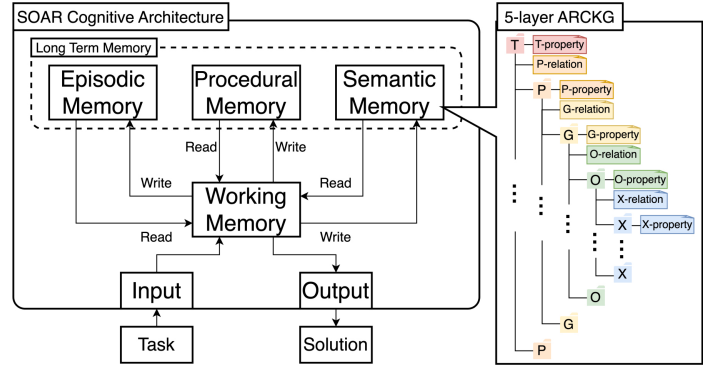


그림 1. ARBOR 아키텍처 개요.

표현되며, 추론이 진행됨에 따라 새로운 사실이 추가되거나 제거된다 [3].

일화 메모리(Episodic Memory)는 풀이 과정에서 시도된 의사결정의 배경, 전후 상황, 순서를 하나의 의사결정 수행도 형태로 기록하도록 설계되어 있다. 유사한 상황에서 과거 경험을 참조하여 탐색을 효율화하는 역할을 목표로 하며, 구체적인 구현은 현재 진행 중이다.

절차 메모리(Procedural Memory)는 의사결정에 사용되는 모든 규칙이 저장되는 장기 메모리로, 초기에 내장된 기본 변환 규칙부터 풀이 과정에서 새로 생성된 규칙까지 포함한다. 각 규칙은 개념명과 해당 개념이 수행하는 변환으로 구성되며, 새로운 문제를 마주했을 때 적용 가능한 규칙이 후보 연산자로 제안된다.

의미 메모리(Semantic Memory)는 과거에 접한 ARC 문제들의 구조적 정보가 영속적으로 저장되는 장기 메모리로, [5]의 접근을 계승하여 각 문제의 구성 요소별 속성과 구성 요소 간 관계가 계층적으로 기록된다. 구체적인 구조는 3.3절에서 설명한다.

3.3 계층적 지식 그래프 ARCKG

ARCKG는 [5]의 4계층 구조를 5계층으로 확장하고 관계 표현 체계를 새롭게 정립한 의미 메모리이다. ARC 문제는 Task(T) → Pair(P) → Grid(G) → Object(O) → Pixel(X)의 5계층으로 분해된다. 물리적 저장은 노드를 폴더로, 엣지를 JSON 파일로 표현하는 파일시스템 기반 구조로 구현되며, 각 노드는 원시 데이터와 속성, 하위 노드들과의 관계 정보를 포함한다.

속성과 관계는 모두 지식 그래프의 엣지에 해당한다. 속성은 0차 엣지로 노드 자신을 가리키며 계층마다 다르게 정의된다(예: 픽셀의 위치, 객체의 색상, 그리드의 크기). 동계층 관계는 같은 계층의 두 노드 속성을 비교하여 COMM과 DIFF를 기록한 1차 엣지로, 불필요한 병목을 방지하기 위해 풀이 과정에서 선택적으로 생성된다. 이계층 관계는 파일 생성을 생략하고 폴더의 종속구조로 대체한다.

비교는 기호적으로 표현된 구조 간에 요소별로 수행된다. 같으면 COMM, 다르면 DIFF로 분류되고 두 노드 각각의 값이 함께 기록된다. 전체 요소 중 COMM 비율은 점수로 기록되어 유사도 지표로 활용되며, 비교 결과는 두 노드의 최소 공통 부모 노드 안에 저장된다. 이 비교 로직은 1차 엣지를 비교 대상으로 확장하면 2차 이상의 엣지

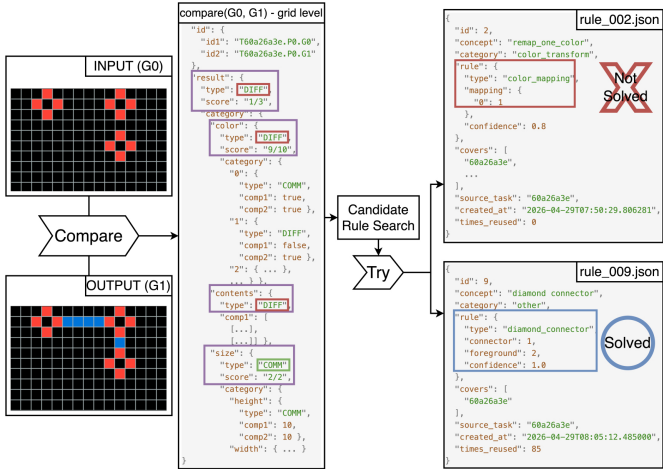


그림 2. ARCKG 비교 및 규칙 적용 예시. 입출력 그리드(G0, G1) 비교를 통해 COMM/DIFF가 기록되고(초록: COMM, 빨강: DIFF), 절차 메모리의 후보 규칙이 순차적으로 시도된다. rule_002는 실패하고 rule_009가 문제를 해결한다.

를 생성할 수 있어, 구조적 패턴 간의 유사성을 포착하는 것이 가능하다. 그림 2는 실제 ARC 문제(60a26a3e)에서 입출력 그리드를 비교한 결과와, 해당 비교로부터 후보 규칙이 시도되는 과정을 보여준다.

3.4 문제 풀이 흐름

ARBOR의 문제 풀이는 두 개의 루프가 중첩된 구조로 진행된다. 안쪽 루프는 문제를 계층적으로 분해하며 패턴을 추출하는 과정이다. 문제가 작업 메모리에 적재되면 현재 수준에서 비교 가능한 정보가 있는지 확인하고, 충분하지 않으면 충돌 상태가 발생하여 한 계층 아래로 내려가 속성을 파악하고 COMM/DIFF를 기록한다. 목표가 설정되어도 현재 정보만으로 달성이 불가능하면 더 깊은 계층으로 내려가고 이것이 반복되며 각 계층의 패턴이 누적된다.

바깥쪽 루프는 그림 2에서 볼 수 있듯이 추출된 패턴을 바탕으로 변환을 탐색하는 과정이다. 의미 메모리에서 유사한 기존 문제를 계층적으로 탐색하고, 절차 메모리에서 후보 규칙을 가져와 현재 문제에 적용한다. DIFF가 해소되면 규칙을 채택하고 풀이를 완료하며, 해소되지 않으면 다음 후보를 시도한다. 후보가 소진되면 안쪽 루프로 돌아가 더 깊은 계층의 패턴을 추가로 추출한다. 두 루프 모두 COMM/DIFF 기반의 명시적 조건 검사가 탐색 방향을 결정하므로, 계층이 깊어질수록 탐색 공간이 누적적으로 축소된다. 풀이에 성공하면 사용된 규칙이 절차 메모리에 저장되고, 풀이 과정 전체가 일화 메모리에 에피소드로 기록된다.

4. 실험

4.1 실험 설계

실험은 두 단계로 구성된다. 1단계는 LLM 기반 자율 개발 루프를 활용하여 ARC 학습 셋 문제들에 대해 절차 메모리에 규칙을 생성하고 축적하는 도움달기 단계이다. 생성된 규칙은 ARBOR가 자율적으로 도출한 것이 아니라 LLM이 각 문제에 대응하여 추가한 결과이다. 2단계는 LLM 없이 ARBOR 단독으로 1,000개의 ARC 학습 셋 문제를 풀이 시도하는 단계로, 1단계에서 활용된 문제도 포함된다.

4.2 결과 및 해석

1단계에서 ARBOR는 40개의 문제를 해결하며 82개의 규칙을 절차 메모리에 축적하였다. 2단계에서 이 82개의 규칙만을 활용하여 1,000개의 문제를 시도한 결과, 72개의 문제를 해결하였다. 규칙 커버리지는 $72 / 82 \approx 0.88$ 이며, 해결된 72개 중 32개는 1단계에서 다루지 않은 새로운 문제로, 축적된 규칙이 일정 수준의 새로운 문제에도 적용되었음을 보인다. 커버리지 1 미만이라는 사실은 분모가 여전히 분자보다 큰 상태임을 의미하며, 커버리지를 1 이상으로 높이기 위해 기존 규칙들의 공통 구조를 자동으로 추출하여 더 일반적인 규칙으로 통합하는 내부 메커니즘이 필요하다.

5. 논의 및 향후 과제

본 논문은 Chollet의 기술 습득 효율 정의에 기반하여 세 가지 설계 조건을 도출하고, 이를 구현한 기호적 문제 해결 시스템 ARBOR를 제안하였다. 초기 실험에서 규칙 커버리지 0.88을 달성하며 규칙 기반 접근의 가능성을 확인하였으나, 설계 목표인 1 이상에는 미치지 못한다.

현재 구현의 핵심 한계는 두 가지이다. 첫째, 절차 메모리의 규칙이 조건부 없이 개념과 변환만으로 구성되어 있어 적용 조건의 명시적 판단이 어렵다. 둘째, 일화 메모리의 구현이 미흡하여 과거 풀이 경험이 새로운 문제에 충분히 활용되지 못하고 있다.

향후 핵심 과제는 반-통합 [7] 기반 자동 일반화 루프의 완성이다. 두 규칙의 공통 골격을 추출하여 더 일반적인 추상 규칙으로 통합하는 메커니즘을 절차 메모리의 저장 직후 단계로 통합함으로써, 규칙 커버리지를 1 이상으로 끌어올리는 것이 목표이다. 아울러 조건-행동 구조를 갖춘 완전한 규칙 체계의 정립과, 일화 메모리를 활용한 경험 기반 탐색 안내 메커니즘의 개발이 함께 필요하다. 장기적으로 ARBOR는 ARC를 넘어 다양한 추상 추론 문제로 확장 가능한 구조를 지향한다.

참고 문헌

- [1] F. Chollet, "On the Measure of Intelligence," arXiv:1911.01547, 2019.
- [2] ARC Prize Foundation, "ARC Prize Leaderboard," arcprize.org/leaderboard, accessed Apr. 2026.
- [3] J. E. Laird, A. Newell, and P. S. Rosenbloom, "SOAR: An architecture for general intelligence," *Artif. Intell.*, vol. 33, no. 1, pp. 1-64, 1987.
- [4] Y. Xu, E. B. Khalil, and S. Sanner, "Graphs, Constraints, and Search for the Abstraction and Reasoning Corpus," in *Proc. AAAI*, 2023.
- [5] M. Lim, S. Lee, L. W. Abitew, and S. Kim, "Abductive Symbolic Solver on Abstraction and Reasoning Corpus," in *Proc. LNSAI@IJCAI*, 2024.
- [6] K. Ellis et al., "DreamCoder: Bootstrapping inductive program synthesis with wake-sleep library learning," in *Proc. PLDI*, 2021, pp. 835-850.
- [7] G. D. Plotkin, "A note on inductive generalization," in *Machine Intelligence 5*. Edinburgh Univ. Press, 1970, pp. 153-163.