진화하는 그래프 구조 학습을 위한 부스티드 DNA 컴퓨팅

석호식[O] 장병탁
바이오지능 연구실, 컴퓨터공학부, 서울대학교
{hsseok[O], btzhang}@bi.snu.ac.kr

Boosted DNA Computing for Evolutionary Graphical Structure Learning

Ho-Sik Seok[0] Byoung-Tak Zhang
Biointelligence lab, School of Computer Science and Engineering, Seoul National University

요 약

DNA 컴퓨팅은 분자 수준(molecular level)에서 연산을 수행한다. 따라서 일반적인 실리콘 기반의 컴퓨터에서와는 달리, 순차적인 연산 제어를 보장하기 어렵다는 특징이 있다. 그러나 DNA 컴퓨팅은 화학반응에 기초한 연산이기 때문에, 실험자가 의도한 연산을 많은 수의 분자에 동시에 적용할 수 있으므로 실리콘 기반의 컴퓨터와는 비교할 수 없는 병렬 연산을 구현할 수 있다. 병렬 연산을 구현하고자 할 때, 일반적으로 연산에 사용하는 모든 DNA 분자들을 대상으로 연산을 구현할 수도 있다. 그러나 전체가 아닌 일부의 분자들을 상대로 연산을 수행하는 것 역시 가능하며 이 때 자연스러운 방법으로 사용할 수 있는 방법이 배깅(Bagging)이나 부스팅(Boosting)과 같은 앙상블(ensemble) 계열의 학습 방법이다. 일반적인 부스팅과 달리 가중치를 부여하는 것이 아니라 특정 학습자(learner)를 나타내는 분자들을 증폭한다면 가중치를 분자의 양으로 표현하는 것이 가능하므로 분자 수준에서 앙상블 계열의 학습을 구현하는 것이 가능하다. 본 논문에서는 앙상블 계열의 학습 방법 중 특히 부스팅의 효과를 DNA 컴퓨팅에 응용하고자 할 때, 어떤 방법이 가능하며, 표현 과정에서 고려해야 할 사항은 어떠한 것들이 있는지 고려하고자 한다. 본 논문에서는 규모를 사전에 한정할 수 없는 진화 가능한 그래프 구조(evolutionary graph structure)를 학습할 수 있는 방법을 찾아보고자 한다. 진화 가능한 그래프 구조는 기존의 DNA 컴퓨팅 방법으로는 학습할 수 없는 문제이다. 그러나 조합 가능한 수를 사전에 정의할 수 없기 때문에 분자의 수에 상관없이 동일한 연산 시간에 문제를 해결할 수 있는 DNA 컴퓨팅의 장점을 가장 잘 발휘할 수 있는 문제이기도 하다.

1. 서 론

DNA 컴퓨팅[1]은 분자 기반의 컴퓨팅이라는 장점 때문에 기존의 실리콘 기반의 컴퓨팅(*in silico* computation)과는 비교할 수 없는 병렬 연산이 가능하다. 따라서 해밀톤 경로 문제(Hamiltonian path problem)와 같은 NP문제를 해결하는데 가능성을 보이고 있다. 그러나 DNA 컴퓨팅은 몇 가지 해결 필요가 있는 단점을 지니고 있다. 첫째, 기술의 문제라는 한계가 있기는 하지만 DNA 컴퓨팅으로 해결하고자 하는 문제가 소위 장난감 문제(toy problem)에 그치고 있다. 둘째, 장난감 문제만을 해결하다 보니 기존의 컴퓨팅 방법(*in silico* computation)에 비해 어떤 장점이 있는지 주장하기가 어렵다. 본 논문에서는 이와 같은 문제를 해결할 수 있는 새로운 응용 분야로써 그래프 구조 학습 문제를 제시한다. 본 논문에서는 그래프 구조 학습을 DNA 컴퓨팅 방법으로 해결하고자 할 때 그래프 구조 표현 및 구조 학습에 사용할 수 있는 방법에는 어떤 것이 있는지 살펴보고자 한다.

2. 그래프 구조 학습 문제

그림 1에서 본 논문에서 해결하고자 하는 그래프 구조 학습 문제를 보이고 있다. 그래피컬 모델(Graphical model)등에서 다루는 그래프와 그림 1에서 보이는 그래프의 차이는 우선 에지의 종류에 있다. 그림 1에서 알 수 있듯 에지에는 "→"와 "●-"의 두 가지 종류가 있다.

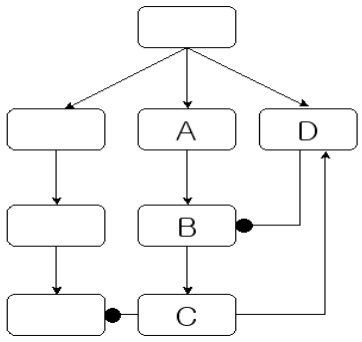

그림 1 → 메시지 전달 경로
●-노드 활동 억제

"→"는 화살표가 시작하는 노드에서 화살표가 향하는 노드로의 메시지 전달을 의미하는 에지로써, 일반적인 그래피컬 모델에서 사용하는 에지와 유사한 의미를 갖는다. "●-"는 새롭게 소개되는 에지로써 "●"가 붙은 노드의 활동을 억제함을 의미한다. 그림 1의 그래프에서 "A"~"D"까지의 노드를 예로 들어 설명하면, 그림 1의 상황은 "노드 A에 메시지 전달->노드 A 활성화->노드 B 활성화->노드 C활성화->노드 D활성화->노드 B 활동 억제"의 활동이 이루어짐을 의미한다. 메시지 전달 방향만 표시하는 일반적인 그래피컬 모델의 그래프와 달리 노드 억제 에지를 추가한 이유는 생물정보학

(bioinformatics)에서 등장하는 각종 신호 전달 경로를 표현하기 위해서이다.

3. 그래프 구조 학습 기제

3.1 DNA 컴퓨팅 학습 기제

일반적으로 DNA 컴퓨팅에서는 각종 문제를 분자 수준(molecular level)의 병렬성을 이용하여 해결하는 접근법을 주로 연구하고 있다[2,3]. 기본적으로 DNA 컴퓨팅 기법에서 제공하는 연산자가 매우 원시적이며 그 용례 또한 제한적이기 때문에 기존의 기계 학습 분야에서 사용하는 각종 고급 학습 기법을 DNA 컴퓨팅에 적용하는 것은 무리가 있다. 기존의 DNA 컴퓨팅 연산자를 학습 기법에 적용하기 위한 시도 중 하나가 바로 PLM (Probabilistic Library Model)이다[4].

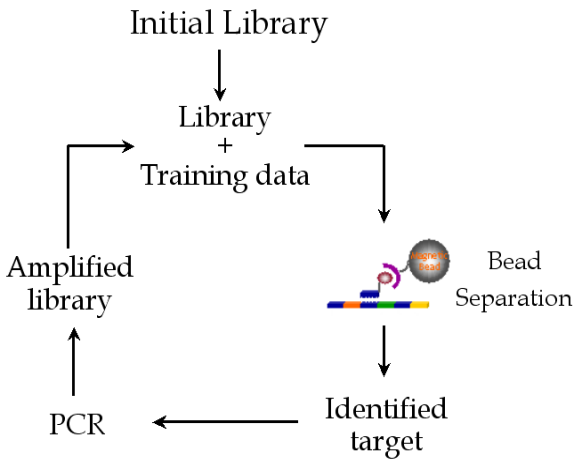

그림 2 PLM 학습 과정

PLM은 DNA 컴퓨팅에서 주로 사용하는 연산자 중 하나인 비드 분리(bead seperation)와 PCR (Polymerase Chain Reaction)을 사용하여 훈련 데이터와 유사한 서열을 갖는 라이브러리 요소를 증폭하여 라이브러리 분포를 변경하는 것을 주요 학습 기제로 사용한다. 즉 PLM에서의 학습은 훈련 데이터의 습득을 통한 라이브러리 분포 변화를 의미하는 것이다. PLM이 DNA 컴퓨팅에서 학습을 진행할 수 있는 한 가지 방법을 제시하기는 하지만, 계속하여 공격받아온 DNA 컴퓨팅의 약점과 관련하여 PLM은 새로운 해결책을 제시해주지는 못한다. 그림 3과 같이 PLM의 라이브러리 요소는 염기 서열의 1차원 구조로 이루어져 있다. PLM에서는 해결하고자 하는 문제를 표현할 수 있는 최대 염기 서열의 수를 사전에 결정한 후, 이들 염기 서열이 만들어 낼 수 있는 모든 패턴의 수를 학습에 사용한다.

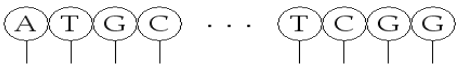

그림 3 PLM의 라이브러리 요소의 최대 크기는 사전에 고정

모든 패턴을 만들어 낼 수 있고 생성한 DNA 서열에 동시에 각종 연산자를 적용할 수 있다는 것이 PLM의 장점이지만 동시에 최대 염기의 수를 사전에 결정한다는 것이 PLM의 단점이기도 하다.

3.2 그래프 구조의 학습

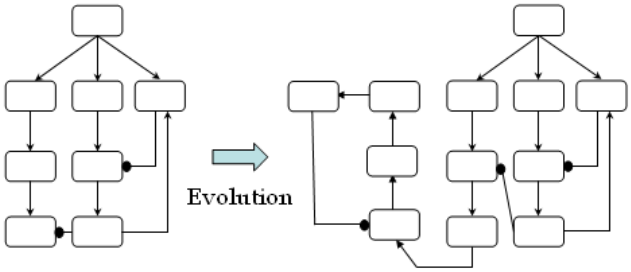

그림 4 진화 가능한 그래프 구조

그림 4에서 진화 가능한 그래프 구조(evolutionary graph structure)를 보이고 있다. 그림 4에서 알 수 있듯, 본 논문에서 소개하고자 하는 진화 가능한 그래프 구조는 진화의 결과에 의해 그래프의 구조가 변화할 수 있는 그래프이다. 기존의 그래피컬 모델과 차이는 노드의 종류 및 노드의 수에 제약이 없다는 점이다. 즉 그림 4에서와 같이 진화 결과 그래프의 크기가 증가하게 되는 것도 가능하다는 것이 진화 가능한 그래프 구조의 특징이다. 사전에 표현 가능한 최대 구조를 설정하고 학습을 진행하는 PLM의 경우, 3.1에서 지적하였듯이 그림 4와 같이 진화를 통해 크기가 증가하는 분야의 학습에 대해서는 학습 방법을 제공하지 못한다는 단점이 있다.

표 1 그래프 구성 요소의 표현 조건

노드: 활성화, 비활성화 상태 표현 가능
에지: 활성화된 노드의 부산물

표1에서는 그래프 구성 요소의 표현 조건을 보이고 있다. 기존 DNA 컴퓨팅 모델에서는 노드와 에지를 모두 염기 서열로 표현한다. 그런데 본 논문의 진화 가능한 그래프 구조에서는 활성화와 억제라는 개념이 존재하므

로 노드와 에지에서도 이들 두 개념을 반영할 필요가 있다. 따라서 노드는 활성화와 비활성화 상태를 표현할 수 있어야 하며, 에지는 활성화된 노드의 부산물로써 다른 노드의 활동에 영향을 미칠 수 있는 형태가 되어야 한다. 그래프 구조 진화를 위해 다양한 방법을 사용할 수 있겠지만, 부스팅[5]이 한 가지 방법이 될 수 있다. 그래프 구조에 대한 학습이 진행되는 상태에서 그림 1의 노드 A와 노드 B 관계가 훈련 데이터로 주어졌다고 가정해 보자. 이 경우 노드 A와 노드 B를 연결하는 관계 이외에는 주어진 훈련 데이터가 소용이 없게 된다. 따라서 노드 A와 노드 B의 관계를 의미하는 노드 및 에지 정보에만 변화가 발생하게 된다. 물론 이와 같은 방식이 일반적인 앙상블(ensemble) 방식이나 부스팅과는 많은 차이가 있지만, 특정 노드 사이의 관계를 표시하는 정보의 변화를 부스팅에서의 가중치로 해석할 수 있다고 전제하면, DNA 컴퓨팅의 학습 방법을 부스팅으로 해석하는 것도 무리한 접근법만은 아니다.

표 2 그래프 진화 방법

양적 부스팅(Quantitative Boosting): 노드의 수를 증가시키는 것으로 가중치 변화 표현
질적 부스팅(Qualitative Boosting): 에지의 작용을 억제하는 요소나 활성화하는 요소를 사용한 가중치 변화 표현

그러나 여기서 주의할 것은 부스팅과 유사한 학습 방법을 사용하였다고 하여, 학습의 결과로 생성된 그래프가 앙상블 계열의 다수결 결정 방식(weighted majority voting)을 사용하지는 않는다는 사실이다. 그림 1의 그래프에서는 출력 부분이 명시되어 있지 않지만, 훈련 과정에서 명시된 입력 노드로 데이터가 부여되면 출력 노드에서 결과를 획득한다는 것을 반드시 염두에 두어야 한다. 훈련 과정에서 사용할 수 있는 부스팅 모델은 표 2에서와 같이 노드의 수를 증감시키는 양적 부스팅과 에지의 활성화를 조정하는 질적 부스팅을 생각할 수 있다. 어떤 방법을 사용할 것인지는 실제 실험에 사용할 연산자의 성격을 고려해서 결정해야 할 것이나, 네트워크 진화의 관점에서는 양적 부스팅이 보다 자연스럽다. 한 편 실제 생물체내의 각종 네트워크에서 관찰되는 억제 및 활성화 효과를 간주하면 질적 부스팅이 보다 자연스러울 수 있다.

4. 결론 및 추후 연구

본 논문에서는 기존의 DNA 컴퓨팅 접근법이 가지고 있는 한계를 해결할 수 있는 접근법으로 진화 가능한 그래프 구조 학습이라는 개념을 제시하였다. 기존의 DNA 컴퓨팅 방법의 경우 문제의 규모가 사전에 결정되기 때문에 확장성이 매우 부족하다. 따라서 DNA 처리에 필요한 시간 및 사전 설계 시간을 고려했을 때, 기존의 컴퓨팅 방법(*in silico* computation) 특히 FPGA등을 이용한 전용 프로세서의 설계를 통한 접근법에 비해 실행 시간 측면에서 어떤 장점이 있을 수 있는지 강력하게 주장하기 어려웠던 것이 사실이다. 그러나 사전에 규모를 결정할 수 없는 진화 가능한 그래프와 같은 문제의 경우, DNA 컴퓨팅의 연산자를 사용할 경우 문제의 규모에 상관없이 동일한 연산시간을 보장받을 수 있으므로 기존의 컴퓨팅 방법이 제공할 수 없는 계산상의 이점을 보장받을 수 있다. 본 논문에서는 기계 학습 방법 중 부스팅을 이용하여 진화 가능한 그래프를 학습할 수 있는 방법을 모색하여 보았다. 본 논문에서는 아이디어만을 제시했을 뿐, 실제 실험을 수행하지는 않았지만, 여기서 제시한 아이디어를 실현할 수 있다면, 각종 생물학 모델에서 발견되는 신호 전달 경로와 같은 그래프 모델을 모델링하고, 학습에 응용할 수 있는 한 가지 가능성을 제시할 수 있게 될 것이다.

감사의 글: 본 연구는 산업자원부 차세대 신기술 과제 및 과학기술부 국가지정연구실 과제에 의해 지원되었음.

참고문헌

[1] Adleman, L., Molecular Computation of Solutions to Combinational Problem, *Science*, Vol. 266: 1021-1024, 1994

[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 the Eighth International Meeting on DNA Based Computers (DNA 8)*, 251-260, 2002

[3] Zhang, B.-T. and Jang, H.-Y., A Bayesian Algorithm for In Vitro Molecular Evolution of Pattern Classifiers, *Preliminary Proceedings of the Tenth International Meeting on DNA Based Computers (DNA 10)*, 294-303, 2004

[4] Zhang, B.-T. and Jang, H.-Y., Molecular Programming: Evolving Genetic Programs in a Test Tube, *The Genetic and Evolutionary Computation Conference (GECCO 2005)*, 2005 (Accepted)

[5] Freund, Y., Boosting a Weak Learning Algorithm by Majority, *Information and Computation*, Vol. 121(2): 256-285, 1995