

# 복수 염기서열 정렬을 위한 한 유용한 알고리즘

김 진\*, 송 민 동\*\*  
\*건국대학교 전산과학과  
\*\*건국대학교 분자생물학과

An efficient algorithm for multiple sequence alignment

Jin Kim\*, Min-Dong Song\*\*  
\*Dept. of Computer Science, Kon-Kuk University  
\*\*Dept. of Molecular Biology, Kon-Kuk University

3개 이상의 DNA 혹은 단백질의 염기서열을 정렬하는 복수 염기서열 정렬(multiple sequence alignment) 방법은 염기서열들 사이의 전화관계, gene regulation, 단백질의 구조와 기능에 관한 연구에 필수적인 도구이다. 복수 염기서열 정렬 문제는 NP-complete 문제군에 속하며, 이 문제를 해결하기 위하여 가장 유용하게 사용되는 알고리즘으로는 dynamic programming이 있다. Dynamic programming은 주어진 입력 염기서열 군들에 대한 최적의 정렬을 생산할 수 있다. 그러나 dynamic programming의 단점은 오랜 실행시간이 요구되며, 때로는 dynamic programming의 속성 때문에 이 알고리즘을 사용하여도 주어진 입력 염기서열 군들에 대한 최적의 정렬을 얻어내지 못하는 경우가 있다. 본 연구에서는 이러한 dynamic programming의 문제를 해결하기 위하여 genetic algorithm을 복수 염기서열 정렬 문제에 적용하였다. 본 논문에서는 genetic algorithm의 design과 적용 방법을 기술하였다. 본 연구에서 제안된 genetic algorithm을 사용하여 dynamic programming의 단점이었던 오랜 실행시간을 줄일 수 있었으며, dynamic programming이 제공하지 못하는 최적의 염기서열 정렬을 계공할 수 있었다.

## 1 서론

DNA, RNA 및 단백질들의 염기서열(sequence)들은 유전에 관한 중요한 정보를 가지고 있다. 염기서열은 DNA의 경우 {a, t, g, c}, RNA인 경우 {a, u, g, c}, 단백질의 경우 {W, Y, F, V, L, I, M, K, R, H, Q, E, D, N, G, A, P, T, S, C}의 일파만으로 이루어진 유한한 길이의 스트링으로 정의할 수 있다. 예를 들면 스트링 "gccttaccggagcc"는 DNA의 염기서열(molecular sequence)이라 할 수 있다. 염기서열 순서확인(sequencing)이란 유전 물질로부터 염기서열의 글자를 읽어내는 작업이라 정의할 수 있다.

복수 염기서열 정렬(multiple sequence alignment) 문제는 계산 생물학(computational biology) 내에서도 가장 중요한 문제 중 하나이다. 복수 염기서열 정렬은 염기서열들 간의 element들의 대응 관계를 알아내는 것과 관련이 있다. 복수 염기서열 정렬은 생물학적 데이터 분석에 있어 중요한 작업으로 미지의 염기서열의 확인을 위한 테이터베이스 검색, 유시현 분자구조와 관련된 패턴인식, 염기서열의 기능 및 가능성이 환경에 대한 중요한 정보를 획득하기 위하여 사용된다. 이러한 염기서열의 정렬을 위해 정렬의 양호성(goodness)을 측정할 수 있는 비용함수( $f$ )가 필요하다. 염기서열들의 특장 집합에 대해, 최적 정렬(optimal alignment)은 최소 비용(minimum cost)을 가지는 정렬이다. 낮은 비용을 기준 강렬을 얻기 위해, 각 염기서열들의 element들 사이에 매치(match), 삽입(insertion), 교체(substitution), 삭제(deletion) 등이 사용된다. 다음은 세 개의 염기서열의 정렬의 한 예이다. 그림 1(a)는 정렬되

지 않은 세 개의 염기서열들을 보여준다. 그림 1(b)는 동일한 세 개의 염기서열을 기본 알파벳과 '-'를 사용하여 세 개의 염기서열들 간의 유사성을 보다 잘 나타낼 수 있도록 표현되었다. 그림 1(b)의

1 2 3 4 5 6 7 8	
염기서열 1 t g c t g c t c	t g c t g c t c
염기서열 2 t g a g c t c	t g a - g c t c
염기서열 3 t g a g t c	t g a - g - t c

(그림 1) 복수 염기서열 정렬의 예. (a) 정렬되지 않은 세 개의 염기서열. (b) 정렬된 염기서열, 잘 정렬된 염기서열은 염기서열들 간의 유사성을 잘 나타낸다.

1, 2, 5, 7, 8 열은 동일한 알파벳으로 매치되었으며, 첫 번째 염기서열의 3번째 요소는 고체점을 보여주며, 세 번째 염기서열의 6번째 열은 삭제되었음을 보여주며, 첫 번째 열의 4번째 열은 삽입되었음을 나타낸다. 삽입과 삭제를 표현하기 위해 기본 알파벳이외에 '-'가 사용되었다. 복수 염기서열 정렬 문제를 다음과 같이 정의할 수 있다.

$S_1, S_2, \dots, S_n$ 는 각각 길이  $n_1, n_2, \dots, n_k$ 의  $n$ 개의 입력 염기서열들이며 ( $S_1=S_{11}S_{12} \dots S_{1n_1}, S_2=S_{21}S_{22} \dots S_{2n_2}, \dots, S_n=S_{n1}S_{n2} \dots S_{nn}$ ) 알파벳은  $\Delta \cup \{-\}$ 이며,  $s_{ij} \in \Delta$ 는 1번쩨의 염기서열을 표시하며,  $j$ 번쩨의 element를 나타낸다 ( $1 \leq i \leq k, 1 \leq j \leq n_i$ ). 그러므로 염기서열 정렬은 2차원 배열과 같다

$$S_I = S_{11}S_{12} \dots S_{1n_1}, \\ S_2 = S_{21}S_{22} \dots S_{2n_2}, \\ \dots \\ S_n = S_{n1}S_{n2} \dots S_{nn}$$

$$S_{IJ} = S_{I1}S_{I2} \dots S_{In_I},$$

복수 염기서열 정렬 문제는 정렬에 대한 최적도의 기준이 주어졌을 때,

최적 비용(optimal cost), 혹은 최적 비용에 근접한 비용(near-optimal cost)를 가지는 정렬을 찾는 문제이디 이 문제는 NP-complete 문제군 [1]에 속하는 문제로 알려있다.

이러한 복수 염기서열 정렬문제를 해결하는 기준의 방법은 dynamic programming[2]을 사용하는 것이다. Dynamic programming은 2개의 염기서열의 정렬시 최적 비용을 가진 정렬을 만들어낸다. 그러나 dynamic programming의 높은 time complexity 및 space complexity 때문에, 짓이가 짧은 염기서열에만 실제로 사용될 수 있다. 복수 염기서열 정렬에 사용되는 또 다른 방법은 simulated annealing[3][4]등과 같은 추정 통계적(stochastic)인 최적화 방법을 사용하는 것이다. Simulated annealing 방법[6]이 3개이상의 염기서열 정렬에 사용되어 왔는데, dynamic programming의 단점을 해결할 수 있으나, 이 방법 또한 많은 계산 시간에 요구되며 local minimum에서 빠져 나오지 못할 경우가 많다. 본 논문에서는 genetic algorithm을 사용하여 복수 염기서열 정렬 문제를 해결하는 방법을 제시하도록 한다. genetic algorithm은 복수 염기서열 정렬문제에 적용하여 기존의 방법들보다 짧은 계산시간으로 최적의 비용을 가지는 복수 염기서열 정렬을 찾도록 한다.

## 2. genetic algorithm

genetic algorithm은 자연계의 진화과정을 흡내낸 계산모델로서 NP-complete 문제 군에 속하는 문제들에 대한 최적 혹은 최적에 가까운 solution을 제공하는 알고리즘이다. genetic algorithm은 개체(individual)라 불리는 후보 해법(candidate solution)들의 모집단(population)들을 운영한다(5). 전형적으로, 초기 모집단은 무작위 개체들로 구성된다. 이후, 개체들은 그들의 상태 적합도(fitness)에 기초하여 모집단에서 제거되거나 재생산된다. 새로운 개체들은 기존의 개체들의 모집단에 다양한 연산자(operator)들을 적용하여 생성된다. 개체의 연속적인 모집단을 세대(generation)이라 한다. genetic algorithm에서 사용되는 연산자들은 다음과 같다.

### 2.1 연산자

- 선택(selection). 각 개체들의 적합도(fitness)가 측정된다. 개체들은 그들의 적합도에 따라 재생산된다. 이때 재생산에 적용하는 방식은 문제마다 같지 않다.
- 교배(crossover) 모집단으로부터 두 개의 개체가 선택되며 개체들 내의 심용하는 위치로부터 substring들이 교환된다. 새로이 생성된 개체들이 다음 세대의 모집단에 추가된다.
- 변이(mutation) 변이는 한 개체의 기본 요소를 변화시켜 새로운 개체를 만들어내는 연산자이다. 변이율에 의해 개체내의 각 요소가 변화되는 확률이 조정된다. 새로이 만들어진 개체가 다음 세대의 모집단에 추가된다.

이러한 연산자들을 일정, 혹은 기법적인 확률에 의해 적용하여 조금 더 우수한 세대, 혹은 optimal solution을 찾는다. 진화의 process는 최초의 개체군에서 시작하여 새로운 개체군들을 반복적으로 생성하며 세대교체를 반복하는 일정의 절단에 의한 텁색과정이다. 세대교체시 적합도가 높은 개체들이 다음 세대의 개체군 생성에 더 많은 영향력을 미치우성 인자들을 유진시키도록 한다.

### 2.2 genetic algorithm의 단계

genetic algorithm의 단계는 다음과 같다.

- 무작위 초기 모집단에서 시작한다

- 모집단 내의 각 후보해법에 대해 쇠적해로서의 적합도를 측정한다.
- 후보 해법들에 연산자를 적용하여 새로운 후보 해법을 만들어낸다.
- 새로운 후보 해법을 평가한 후 모집단에 추가한다.
- 총료조건을 만족할 때까지 3~4단계를 반복하고, 만족하면 최적 적합도를 가진 해법을 출력한다.

## 3. 복수 염기서열 정렬을 위한 genetic algorithm

### 3.1 비용 함수

염기서열 정렬에 사용되기 위한 비용 함수(cost function)는 정렬의 질(quality)에 대한 명백한 측정수단이어야 한다. 즉 작은 비용을 가지는 복수염기서열은 큰 비용을 가지는 복수염기서열보다 생물학적 현상을 잘 표현하여야 한다. Altschul[7][8]은 복수 염기서열 정렬을 위한 몇 가지의 비용 함수를 분류하였다. 이들 비용 함수들은 교체 비용(substitution cost)과 갭 비용(gap cost)으로 구성된다.

- 교체 비용 교체 비용은 DNA나 단백질 분자를 대표하는 글자들을 정렬하는데 드는 비용이다.

- 갭 비용. 갭 비용은 갭('-)에 부과되는 비용이다.

어떤 복수 염기서열 정렬 A에 대한 비용 함수 f는  $f(A) = \text{substitution cost} + \text{gap cost}$ 로 정의할 수 있으며, 복수 염기서열 정렬 문제는 최소의 비용을 가지는 염기서열 정렬을 찾는 것을 목표로 하는 문제이다.

### 3.2 초기 모집단의 생성

genetic algorithm의 처음 단계는 초기 모집단의 생성이다. 물론 경험적인 heuristic 알고리즘[9]을 사용하여 최초의 해를 얻는다. 이 해에 들어있는 갭('-)들의 위치를 변경하여 50개체로 구성된 초기 모집단을 생성한다. 이때 만약 dynamic programming에 의해 얻어진 복수 염기서열보다 개선된 복수 염기서열 정렬을 위해서는, dynamic programming에 의해 얻어진 해를 초기 해로 사용할 수도 있다. 이 해에 genetic algorithm을 적용하였을 때에 해가 개선되지 않았을 때 이 해는 최적해라 할 수 있다.

### 3.3 개체의 평가 및 새로운 세대의 생성

세 세대를 만들기 위하여 각 개체의 적합도를 평가한다. 이 적합도는 비용 함수에 따라 각 정렬에 대한 비용을 계산하여 얻는다. 비용이 적은 정렬일수록 적합도는 높아진다. 현 세대의 개체를 기준해 적합도가 낮은 50%만이 후손으로 고려된다.

새로운 후손들은 부모를 선택한 후 그들을 변경하여 생성한다. 부모를 선택할 때 부모의 적합도와 관련된 확률에 의한다. 부모를 변경하기 위하여 몇 가지의 연산자가 사용되었다.

- 변이 연산자 Kim[6]은 simulated annealing을 복수 염기서열 정렬에 적용하였다. 이때 사용된 연산자는 swap 연산자였다. 본 논문에서는 이 연산자를 swap연산자를 변이 연산자로 사용하였다(그림 2). swap연산자는 기본적으로 정렬에 들어있는 갭들의 위치를 달리해주는 연산자이다.

AKNKQIIGGA—MSG	AKNKQIIGGA—MSG
AKMKQIIGGA—MSG	AKMKQIIGGA—MSG
AKMKK—IGGATG	AKMKKIGGAA—TG

(a)

(b)

(그림 2) swap연산자의 예. (a) 기존의 개체. (b) swap연산자에 의해 새롭게 만들어진 새로운 개체

2) 교배 연산자, 교배 연산자는 두 개의 다른 정렬을 혼합하여 하나 혹은 두 개의 새로운 정렬을 만들어낸다 첫 번째 부모 정렬의 한 열이 무작위로 선택되며, 그 열을 기준으로 두 부분(X, Y)으로 나눈다 두 번째 부모 정렬의 대응되는 부분이 역시 두 부분으로 나누어지며(A', B'), 나누어진 부분의 일부를 결합하여(X-Y, X'+Y) 새로운 개체들을 만든다 새로운 개체들의 비용 합수를 측정하여 우수한 것만을 새로운 세대에 삽입시킨다(그림 3)

AKMKQIGGA---MSG	AKMKQI---GGAGMSG
AKMKQIGGA---TG	AKMKQIGG---A TG
AKMKQIGGA---TG	AKMKQIGG---A TG
(a)	(b)
AKMKQIGGA---MSG	AKMKQI---GGAMMSG
AKMKQIGGA---TG	AKMKWIGG---ATG
AKMKQIGGA---TG	AKMKWIGG---ATG
(c)	(d)

(그림 3) 교배 연산자의 예 (a), (b)는 부모 정렬들.

(c), (d)는 교배 연산자에 의해 만들어지는 새로운 개체. |를 기준으로 부모 정렬들은 두 부분으로 나누어진다

genetic algorithm은 추정 통계적인 방법이므로 이론적으로 무한한 계산시간을 적용하여도 이 방법이 최적 비용을 가진 복수 염기서열 정렬을 생성할 수 있다는 보장이 없다. 이것은 다른 추정 통계적인 기법들에도 적용된다. 본 연구에서는 10000세대 이후 프로그램을 종료하도록 하고 그 과정에서 얻어진 최소의 비용을 가진 복수 염기서열 정렬을 출력하도록 하였다

#### 4. 실험 및 결과

본 논문의 알고리즘이 PC 상에서 구현되어 chymotrypsin, trypsin과 elastase family에 속하는 염기서열들에 대한 복수염기서열 정렬을 시도하였다. 정렬한 염기서열의 개수는 3~10개이었으며, 길이는 200~300이었다. 한 세대의 개체 수는 100이었다. 10개의 염기서열을 정렬하는데 드는 시간은 2시간이내였다. (그림 4)은 5개의 염기서열을 정렬한 결과의 일부분을 보였다

IIGGVESIPHSRPMYAHLDIVTEKGLRVCIGGFLISRQFVLTAAHC  
IVGGTNSWGEWPWCVSLQVKLTQQR-HLCCGGSLLIGROWVLTAAC  
IVNGEEAVPGSVPWQVSLSQ---DKTGP-HFCGGSLINNEWWWVITAAC  
IVGGYTGCANTVPGVSLN---SGY-HFCGGSLINSQWVVAAC  
VVGSTEAQRNNSWPQISLQYRGSSSWAHTCGGTLLIRQNWWVMTAAC  
VVGTRAQGEPPFMVRSL---MCGCGALYAQQDIVLTAAC  
(a)

IIGGVESIPHSRPMYAHLDIVTEKGLRVCIGGFLISRQFVLTAAHC  
IVGGTNSWGEWPWCVSLQVKLTQQR-HLCCGGSLLIGROWVLTAAC  
IVNGEEAVPGSVPWQVSLSQDKTG---FHFCGGSLINNEWWWVITAAC  
IVGGYTGCANTVPGVSLNNSG---YHFCGGSLINSQWVVAAC  
VVGSTEAQRNNSWPQISLQYRGSSSWAHTCGGTLLIRQNWWVMTAAC  
VVGTRAQGEPPFMVRSL---CGCGALYAQQDIVLTAAC  
(b)

(그림 4) (a) dynamic programming에 의하여 생성된 복수염기서열 정렬 이 정렬이 가진 비용은 9663이다. (b) genetic algorithm에 의하여 생성된 복수염기서열 정렬. 이 정렬이 가진 비용은 9648이다. genetic algorithm이 보다 작은 비용을 가진 복수 염기서열 정렬을 생성한다

genetic algorithm의 장점은 복수 염기서열 정렬에 관한 어떠한 비용 합수라도 사용할 수 있다는 점이다. 부모개체를 선택한 후 연산자들을 적용하여 완벽한 새로운 개체를 만들어내기 때문에 비용 합수를 적용

하는데 필요한 모든 정보를 얻을 수 있다. Dynamic programming의 경우에 있어서는 알고리즘의 한계성 때문에 특정 비용합수를 복수 염기서열 정렬에 사용할 수 없다.

genetic algorithm의 또 다른 장점은 기존의 dynamic programming보다 적은 계산시간을 요구한다는 것이다. 정렬하려는 염기서열의 개수가 많아지면 이 계산시간의 차이점도 증가한다.

본 연구에서는 두 개의 가장 기본적인 연산자를만이 사용되었다 그러나 이 두 개의 기본 연산자들 이외에도 보다 효율적인 연산자들이 있을 수 있다. 본 연구에서는 경험적인(heuristic) 알고리즘에 의해 만들어진 해를 초기 해의 원천으로 사용하기 때문에 local minimum에서 빠져 나오지 못할 수 있다. 이를 해결하기 위한 인산자가 필수적이다

#### 5. 결론

본 논문에서는 genetic algorithm 알고리즘을 사용하여 3-10개의 최소 비용을 가지는 복수 염기서열 정렬을 구하는 방법을 소개하였다. 이 방법은 기존의 알고리즘이나 빠르게 최소 비용에 가까운 값을 가지는 복수 염기서열 정렬을 생성할 수 있었다. 현재 이 알고리즘을 Visual C++를 사용하여 PC상에서 구현하여 생물학자들에게 보다 쉽게 사용할 수 있도록 구현 중에 있다.

#### 참고문헌

- [1] M R Garey and D. S. Johnson "Computers and Intractability A guide to the theory of NP-completeness", W H Freeman, San Francisco, CA , 1979
- [2] S B Needleman and C D Wunch "A general method applicable to the search for similarities in the amino acid sequences of two proteins", J. Mol Biol. Vol 48, pp 443-453, 1970
- [3] M Metropolis, M Rosenbluth, A. Rosenbluth, A. Teller and E. Teller "Equation of state calculations by fast computing machines", J. Chem. Phys. Vol 21, pp 1087-1093, 1953
- [4] S. Kirkpatrick, C. D Gelatt and M P. Vecchi "Optimization by simulated annealing", Science Vol 220, pp.671-680, 1983
- [5] D. E Goldberg. "Genetic algorithms in search, optimization, and machine learning", New York Addison-Wesley, 1989
- [6] J Kim, S Pramanik and M J. Chung "Multiple sequence alignment using simulated annealing", Comp Appl Biosci Vol 10, pp.419-426, 1994
- [7] S. F Altschul. "Gap costs for multiple sequence alignment", Journal of Theor Biol Vol 138, pp 297-309, 1989
- [8] S. F. Altschul and D. J. Lipman. "Trees, stars, and multiple biological sequence alignment", SIAM J Appl. Math Vol 49, pp.153-161, 1989.
- [9] D G Bacon and W. F. Anderson "Multiple sequence alignment", J. Mol. Biol Vol 191, pp 153-161, 1986.

본 연구는 한국학술진흥재단의 '97 공모파제(학제간 연구) 지원으로 수행된 연구결과의 일부임