ISSN(Print): 2287-5905

http://dx.doi.org/10.3745/KTSDE.2013.2.2.119

Approximate Periods of Strings based on Distance Sum for DNA Sequence Analysis

Ju Hui Jeong[†] · Young Ho Kim[†] · Joong Chae Na[††] · Jeong Seop Sim[†††]

ABSTRACT

Repetitive strings such as periods have been studied vigorously in so diverse fields as data compression, computer-assisted music analysis, bioinformatics, and etc. In bioinformatics, periods are highly related to repetitive patterns in DNA sequences so called tandem repeats. In some cases, quite similar but not the same patterns are repeated and thus we need approximate string matching algorithms to study tandem repeats in DNA sequences. In this paper, we propose a new definition of approximate periods of strings based on distance sum. Given two strings $p(|p|=m)$ and $x(|x|=n)$, we propose an algorithm that computes the minimum approximate period distance based on distance sum. Our algorithm runs in $O(mn^2)$ time for the weighted edit distance, and runs in $O(mn)$ time for the edit distance, and runs in $O(n)$ time for the Hamming distance.

Keywords : Approximate Period, Distance Sum, Approximate String Matching

DNA 서열분석을 위한 거리합기반 문자열의 근사주기

정 주 희[†] · 김 영 호[†] · 나 중 채[††] · 심 정 섭[†††]

요 약

주기와 같은 반복문자열에 대한 연구는 데이터압축, 컴퓨터활용 음악분석, 바이오인포매틱스 등 다양한 분야에서 진행되고 있다. 바이오인포매틱스 분야에서 주기는 유전자 서열이 반복적으로 나타나는 종렬중복과 밀접한 관련이 있으며 이는 근사문자열매칭을 이용한 근사주기 연구와 관련이 있다. 본 논문에서는 기존의 근사주기에 대한 정의를 보완하는 거리합기반 근사주기를 정의하고 이에 대한 연구 결과를 제시한다. 길이가 각각 m과 n인 문자열 p와 x가 주어졌을 때, p의 x에 대한 거리합기반 최소 근사주기거리를 가중편집거리에 대해 $O(mn^2)$ 시간, 편집거리에 대해 $O(mn)$ 시간, 해밍거리에 대해 $O(n)$ 시간에 계산하는 알고리즘을 제시한다.

키워드 : 근사주기, 거리합, 근사문자열매칭

1. 서 론

문자열매칭(string matching)알고리즘은 많은 분야에서 활발히 연구되고 있다. 특히 주기(period)와 같은 반복문자열에 대한 연구는 데이터압축, 컴퓨터활용 음악분석, 바이오인포매틱스 등 다양한 분야에서 진행되고 있다[1-3]. 주기에 대한 정의는 다음과 같다. 길이가 각각 m과 n인 문자열 p와 x에 대해 p^k를 p가 k번 반복된 문자열이라 하고 p'을 p의 접두사라 할 때, $x=p^kp'\ (k\geq 1)$이면 p를 x의 주기라 한다. 예를 들어 $x=abaabaab$이면 aba가 x의 주기이다. [4]에는 $O(n)$시간에 문자열의 주기를 찾는 알고리즘이 제시되어 있다.

바이오인포매틱스 분야에서 주기는 유전자 서열이 반복적으로 나타나는 종렬중복(tandem repeats)과 밀접한 관련이 있다. 인간 유전체의 10%는 이러한 종렬중복으로 이루어져 있다고 알려져 있다[6]. 인간의 30억 개의 유전자에서 이러한 반복된 서열이 나타나는 부분에는 인간의 유전적 특성과 관련된 정보들을 담고 있어 유전체 분석에서 중요하게 다루어지고 있다[3,5-7]. 그런데 유전체에서는 정확히 일치하는 서열들만 반복되어 나타나는 것이 아니라, 어느 정도 다르지만 유사한 부분들이 반복되는 경우들이 자주 발생한다.

※ 이 논문은 지식경제부 및 한국산업기술평가관리원의 IT산업원천 기술개발산업의 일환으로 수행하였음.[10038768, 유전체 분석용 슈퍼컴퓨팅 시스템 개발]
※ 이 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No.2012R1A2A2A01014892, No. 2012-0003214)
※ This work was supported by the Industrial Strategic technology development program (10041971, Development of Power Efficient High-Performance Multimedia Contents Service Technology using Context-Adapting Distributed Transcoding) funded by the Ministry of Knowledge Economy(MKE, Korea)
※ 이 논문은 인하대학교의 지원에 의하여 연구되었음.
† 준 회 원 : 인하대학교 컴퓨터정보공학과 석사과정
†† 종신회원 : 세종대학교 컴퓨터공학과 교수
††† 종신회원 : 인하대학교 컴퓨터정보공학부 교수
논문접수 : 2013년 1월 8일
수 정 일 : 1차 2013년 1월 24일
심사완료 : 2013년 1월 24일
* Corresponding Author : Jeong Seop Sim(jssim@inha.ac.kr)

이렇게 바이오인포매틱스 분야에서 발생하는 유사한 부분들의 반복에 대한 연구와 관련된 문제들은 근사문자열매칭(approximate string matching)알고리즘을 이용하여 해결할 수 있으며 특히 종렬중복과 관련하여 근사적으로 반복되는 문자열에 대한 연구가 다양하게 진행되고 있다[6-7].

[8]에서는 반복되는 문자열 사이의 불일치를 허용한 근사주기에 대한 연구를 수행하였다. 이때 문자열 사이에 나타나는 불일치 정도를 오차(error) 또는 거리(distance)라 하며 이는 편집거리(edit distance), 가중편집거리(weighted edit distance), 해밍거리(Hamming distance)와 같은 거리함수를 이용하여 계산할 수 있다. [8]에서 제시된 근사주기의 정의는 다음과 같다. 문자열 x와 p 그리고 거리함수가 주어졌을 때 $x = p_1 p_2 \cdots p_r\ (p_i \neq \epsilon, 1 \leq i \leq r)$과 같이 x를 서로 겹치지 않는 p_i들로 나눌 수 있다. $d(a,b)$를 문자열 a, b의 거리라 하고 p'을 p의 접두사라 할 때 $d(p,p_i) \leq t\,(1 \leq i < r)$이고 $d(p',p_r) \leq t$를 만족하면 p를 x의 t-근사주기라고 한다. 이때 각 p_i를 주기블록이라 하고 그러한 t들 중 최소값을 p의 x에 대한 근사주기거리라 한다. [8]에서는 가중편집거리에 대해 $O(mn^2)$시간, 편집거리에 대해 $O(mn)$시간, 해밍거리에 대해 $O(n)$시간에 근사주기거리를 계산하는 알고리즘을 제시하였다.

DNA 서열분석 과정에서 [8]에서 제시된 근사주기의 정의를 이용할 때 바람직하지 않은 경우가 발생할 수 있다. DNA 서열분석 과정 중 염기서열결정과정(sequencing)에서는 실험적 오류가 발생할 수 있다. 예를 들어 어떤 DNA 서열이 $AATAATAAT$였다고 가정해보자. 이 서열에 대해 염기서열결정과정에서 오류가 발생하여 Fig. 1과 같이 $AATTTTAAT$로 결정될 수 있다. 실제 서열에서는 AAT 서열이 세 번 반복되어 있지만 [8]에서 제시된 근사주기의 정의를 이용하면 Fig. 1A의 밑줄 친 부분과 같이 $AATT$와 $TTAAT$로 주기블록이 나누어지게 된다. 이 결과 실제로 오차가 없는 주기블록을 오차가 발생한 블록으로 판단하게 되며 주기의 반복 회수도 3회인 서열을 2회로 잘못 분석하는 결과를 만들 수 있다. 이와 같은 경우는 Fig. 1B와 같이 한 주기블록에서는 큰 오차가 발생하지만 나머지 2개의 주기블록에서는 오차가 없도록 분할하는 것이 더 바람직하다고 할 수 있다.

본 논문에서는 길이가 각각 m과 n인 문자열 p와 x가 주어졌을 때, p의 x에 대한 거리합기반 근사주기를 정의하고 최소 근사주기거리를 찾는 문제를 제시한다. 그리고 가중편집거리에 대해 $O(mn^2)$시간, 편집거리에 대해 $O(mn)$시간, 해밍거리에 대해 $O(n)$시간에 각각 거리합기반 근사주기거리를 계산하는 알고리즘을 제시한다. 본 논문에서 제시한 근사주기 정의 및 해결 알고리즘을 이용하면 Fig. 1B의 밑줄 친 부분과 같이 주기블록이 AAT, TTT, AAT로 나누어지게 되어 Fig. 1A에서와 같은 바람직하지 않은 결과를 보완할 수 있다.

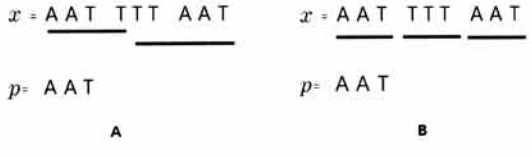

Fig. 1. A. An example for period blocks, B. An example for period blocks based on distance sum

본 논문의 구성은 다음과 같다. 2장에서는 본 논문의 알고리즘을 위한 선행연구에 대해 설명한다. 3장에서는 거리합기반의 근사주기와 근사주기거리를 찾는 문제를 정의하고 이를 해결하는 알고리즘을 제시한다. 그리고 4장에서 결론과 향후연구 방향을 제시한다.

2. 선행 연구

2.1 거리함수

문자열은 알파벳 집합 Σ에 속한 0개 이상의 문자들의 나열이다. 문자열 A의 길이는 $|A|$로 나타내며 $A[i]\,(1 \leq i \leq |A|)$는 A의 i번째 문자를 나타낸다. 또한 A의 $A[i]A[i+1]..A[j]$는 $A[i..j]$로 표기한다. 그리고 $\triangle$는 공백(empty)문자를 나타낸다.

문자열 A와 B에 대해서 거리함수 $d(A,B)$는 A를 B로 변환하는데 필요한 최소 비용을 나타낸다. 대표적인 거리함수로는 편집거리, 가중편집거리, 해밍거리가 있다[9]. 편집거리는 A를 B로 변환하는데 필요한 최소 편집연산의 수이다. 이때 편집연산은 삽입, 삭제, 교체연산으로 구성된다. 해밍거리는 $|A| = |B|$일 때 A를 B로 변환하는데 필요한 최소 교체연산의 수이다. 가중편집거리는 모든 문자쌍에 대해 삽입, 삭제, 교체비용을 나타낸 비용행렬(penalty matrix)이 주어졌을 때 A를 B로 변환하기 위해 필요한 최소 비용이다. 만약 모든 문자 $a,b,c \in \Sigma$에 대해 다음 네 가지 조건을 만족하면, 비용행렬을 메트릭(metric)이라고 한다.

$$d(a,b) \geq 0$$
$$d(a,b) = d(b,a)$$
$$d(a,a) = 0$$
$$d(a,c) \leq d(a,b) + d(b,c)$$

Fig. 2는 메트릭인 비용행렬에 대한 예이다. 가중편집거리는 Fig. 2와 같이 비용행렬이 주어졌을 때 비용행렬에 저장된 값들을 참조하여 두 문자열 사이의 거리를 계산한다. 편집거리는 각 문자쌍 사이의 거리가 모두 1인 경우이다.

	a	b	c	$\triangle$
a	0	2	1	2
b	2	0	2	1
c	1	2	0	1
$\triangle$	2	1	1	0

Fig. 2. A penalty matrix

2.2 D 테이블

두 문자열 A와 B가 주어졌을 때 잘 알려진 동적프로그래밍 기법을 이용하여 편집거리 및 가중편집거리를 계산할 수 있다. 이때 계산된 $(|A|+1) \times (|B|+1)$크기의 테이블을 D 테이블이라 하자.

D 테이블을 초기화 하는 방법은 다음과 같다.

$$\begin{pmatrix} D[0,0] = 0, \\ D[i,0] = D[i-1,0] + d(A[i], \triangle)\ (1 \leq i \leq |A|), \\ D[0,j] = D[0,j-1] + d(\triangle, B[j])\ (1 \leq j \leq |B|) \end{pmatrix} \quad (1)$$

각 $D[i,j](1 \le i \le |A|, 1 \le j \le |B|)$는 식 (2)에 의해 계산된다.

$$D[i,j] = \min \begin{pmatrix} D[i-1,j]+d(A[i],\Delta), \\ D[i,j-1]+d(\Delta,B[j]), \\ D[i-1,j-1]+d(A[i],B[j]) \end{pmatrix} \quad (2)$$

$D[i,j]$는 $A[1..i]$를 $B[1..j]$로 변환하는데 필요한 최소의 비용을 나타낸다. 따라서 $D[|A|,j](1 \le j \le |B|)$에 저장된 값은 A와 $B[1..j]$의 편집거리 혹은 가중편집거리이다. 즉, A와 B의 접미사의 편집거리 혹은 가중편집거리를 계산하기 위해서 D 테이블을 새로 계산할 필요가 없다. 그리고 $D[|A|,|B|]$가 A와 B의 편집거리 또는 가중편집거리를 의미한다.

	△	b	b	a	b	a
△	0	1	2	3	4	5
a	1	1	2	2	3	4
b	2	1	1	2	2	3
a	3	2	2	1	2	2
b	4	3	2	2	1	2

Fig. 3. Computation of edit distance between *abab* and *bbaba* using D table

예를 들어 Fig. 3은 *abab*와 *bbaba*의 편집거리를 계산하기 위한 D 테이블이다. $D[4,4]$의 1은 A와 $B[1..4]$의 편집거리이며 $D[4,5]$의 2는 A와 B의 편집거리이다. 식 (1)과 (2)에 의해 각 $D[i,j]$는 $O(1)$에 계산되므로 D 테이블은 $O(|A||B|)$ 시간에 계산된다[9].

3. 거리합기반 문자열의 최소 근사주기거리 찾기

3.1 거리합기반 근사주기와 알고리즘

본 논문에서 제시하는 거리합기반 근사주기는 다음과 같이 정의한다.

정의 1. 거리합기반 근사주기와 근사주기거리

x가 $x = p_1 p_2 \cdots p_r (p_i \neq \epsilon, 1 \le i \le r)$과 같이 p_i로 나누어질 수 있고 p'을 p의 접두사라고 하자. 이때 $\sum_{i=1}^{r-1} d(p,p_i) + d(p',p_r) \le s$이면 p를 x의 거리합기반 s-근사주기 또는 거리합이 s인 근사주기라고 정의한다. 이때 각 p_i를 주기블록이라고 하며 s 중 최소값을 p의 x에 대한 거리합기반 근사주기거리라고 한다. 앞으로는 이를 근사주기거리라고 표현한다.

거리합기반 근사주기거리를 찾는 문제를 다음과 같이 정의할 수 있다.

문제 1. 거리합기반 문자열의 근사주기거리 찾기

입력: 길이가 각각 m과 n인 문자열 p와 x, 거리함수 d

출력: p의 x에 대한 거리합기반 최소 근사주기거리 s

위의 거리합기반 근사주기거리 찾기 문제는 [8]에 제시된 알고리즘을 변형하여 해결할 수 있다. 본 논문에서 제시한 근사주기거리 찾기 알고리즘은 p와 x의 부분문자열 사이의 거리를 계산하는 단계와 실제 최소 근사주기거리 s를 계산하는 단계로 구성된다.

단계 1. p와 x의 부분문자열 사이의 거리 계산

w_{ij}를 p와 x의 부분문자열 $x[i..j](1 \le i \le j < n)$사이의 거리, w_{in}을 p의 접두사들과 $x[i..n]$사이의 거리의 최소값으로 정의한다. 이 값들을 계산하기 위해 p와 x의 모든 접미사들 사이의 D 테이블을 생성한다. 생성된 p와 $x[i..n]$사이의 D 테이블에서 $D[m,j]$가 w_{ij}를 나타낸다. 또한 w_{in}은 n번째 열에서 최소값이 된다. 단계1에서 계산된 값들은 단계2에서 근사주기거리를 계산하는데 사용된다.

단계 2. 거리합기반 근사주기거리 계산

단계2에서는 p의 x에 대한 거리합기반 근사주기거리 s를 동적프로그래밍 기법을 이용하여 계산한다. 총 n 번의 계산과정으로 이루어져있으며 각 계산과정마다 s_i를 계산한다. s_i는 p의 $x[1..i](1 \le i \le n)$에 대한 근사주기거리를 나타낸다. $s_0 = 0$으로 초기화하고 $i = 1$부터 n까지 차례로 식 (3)을 이용하여 s_i값을 계산한다.

$$s_i = \min_{0 \le h < i}(s_h + w_{h+1,i}) \quad (3)$$

문자열은 다양한 크기의 주기블록으로 나눌 수 있으며 주기블록을 나누는 방법에 따라 계산되는 s_i가 달라진다. 그러므로 식 (3)에서 0부터 $i-1$까지 주기블록을 나누는 위치를 변경하여 s_i를 계산하고 이 중 최소값을 찾아 근사주기거리를 계산한다. 이때 h는 s_h와 $w_{h+1,i}$의 합을 최소로 만드는 위치이다. 즉, p의 $x[1..i](1 \le i \le n)$에 대한 근사주기거리를 찾고 주기블록들을 결정했을 때, h는 마지막 주기블록의 시작 위치를 나타낸다. 마지막으로 계산된 s_n이 p의 x에 대한 거리합기반 근사주기거리가 된다. 예를 들어 x가 $GATGAATGAA$일 때 근사주기거리는 3이 되고 이것은 x가 주기블록 $GAT, GAAT, GAA$로 나누어 졌을 때 각 주기블록들과 p와의 거리의 합을 의미한다. 이는 주기블록을 나누는 여러 방법 중 최소 근사주기거리를 계산한다.

3.2 시간복잡도 분석

거리함수로 편집거리, 가중편집거리, 해밍거리를 사용할 때 알고리즘의 수행시간은 다음과 같다.

먼저 메트릭인 가중편집거리를 사용했을 때 알고리즘의 각 단계별 수행시간을 고려해 보자. 단계1에서 p와 x의 모든 접미사들 사이의 D 테이블을 생성하므로 총 n개의 D 테이블이 생성된다. 따라서 단계1을 계산하는데 $O(mn^2)$시간이 필요하다. 그리고 단계2에서 각 s_i를 계산할 때 h에 따라 $O(n)$번 비교를 하므로 $O(n^2)$시간이 필요하다. 따라서 거리합기반 근사주기거리를 계산하기 위해서는 총 $O(mn^2)$ 시간이 필요하다. 가중편집거리에서 사용되는 비용행렬에서 최대 가중치가 상수일 경우에는 [11]에 제시된 알고리즘을 이용하여 개선할 수 있다. [11]에는 p와 $x[1..n]$ 사이의 가중편집거리를 알고 있을 때 p와 $x[2..n]$의 가중편집거리를 $O(\min\{c(m+n), mn\})$시간에 계산하는 알고리즘이 제시되어 있다. 이때 c는 최대 가중치를 나타낸다. 만약 c가 상수라면 시간복잡도는 $O(m+n)$이 되므로 단계1

에서 n개의 D 테이블을 생성하는 시간은 $O(n^2)$이 된다. 따라서 가중편집거리를 사용했을 때 $O(n^2)$시간에 근사주기거리를 계산할 수 있다.

두 번째로 편집거리를 사용할 때 알고리즘의 시간복잡도를 계산해보자. [6]의 편집거리에 대해 근사주기거리를 찾는 알고리즘은 다음과 같은 속성을 만족한다. p와 주기블록 사이의 거리가 m보다 크면 주기블록을 둘로 나누었을 때 거리가 더 작아진다. 따라서 근사주기거리를 계산할 때 p와 길이가 $2m$을 넘지 않는 x의 부분문자열들만 고려한다. 이러한 속성은 거리합기반 근사주기거리를 계산할 때는 만족하지 않을 수 있지만 p와 주기블록 사이의 거리가 m보다 큰 경우 p와 일치하지 않는 문자의 수가 $|p|$보다 많다는 것을 의미하므로 반복되는 문자열이라고 보기 어렵다. 따라서 본 논문에서는 p와 주기블록 사이의 거리를 m으로 제한하였다. 그러므로 본 논문에서도 편집거리에 대해 근사주기거리를 계산할 때 길이가 $2m$을 넘지 않는 x의 부분문자열만 고려한다. 이때 각 단계별 시간복잡도는 다음과 같다. 단계1에서 크기가 $(m+1)\times(2m+1)$인 D 테이블이 n개 생성되므로 $O(m^2n)$시간이 필요하다. 또한 단계2에서 각 s_i를 계산할 때 길이가 $2m$을 넘지 않는 x의 부분문자열만 고려하므로 $O(m)$ 번 비교한다. 총 n 번의 계산과정이 필요하므로 단계2는 $O(mn)$시간에 계산된다. 따라서 편집거리를 사용했을 때 $O(m^2n)$시간에 근사주기거리를 계산할 수 있다. 한편 [12]의 결과를 이용하면 1단계에 필요한 시간을 $O(mn)$으로 개선할 수 있다. [12]를 통해 p와 $x[1..2m]$의 거리에 대한 해를 이용하여 p와 $x[2..2m+1]$의 거리에 대한 해를 $O(m)$시간에 계산할 수 있다. 그러므로 단계1을 $O(mn)$시간에 해결할 수 있게 된다. 따라서 편집거리를 사용했을 때 $O(mn)$시간에 근사주기거리를 계산할 수 있다.

세 번째로 해밍거리를 사용했을 때 알고리즘의 시간복잡도를 계산해보자. 해밍거리는 비교하는 두 문자열간의 길이가 같아야 한다. 따라서 단순 문자비교를 이용하여 근사주기거리를 계산할 수 있으므로 해밍거리를 사용했을 때 $O(n)$시간에 근사주기거리를 계산할 수 있다.

4. 결 론

본 논문에서 제시한 거리합기반 근사주기의 정의는 기존의 근사주기를 이용했을 때 발생할 수 있는 오류를 보완할 수 있다. 즉, 다른 주기블록에 비해 하나의 주기블록에 많은 오차가 생겼을 때 이로 인해 주기블록을 제대로 분할하지 못할 수 있는 경우를 해결할 수 있다. 따라서 향후 기존의 근사주기의 정의와 본 연구에서 제시된 거리합기반의 근사주기의 정의를 함께 고려한다면 바이오인포매틱스 분야에서 반복적인 서열에 대한 좀 더 의미 있는 분석이 가능할 것으로 기대된다.

참 고 문 헌

[1] Y. Lifshits, "Solving classical string problems an compressed texts," In Combinatorial and Algorithmic Foundations of Pattern and Association Discovery, number 06201 in Dagstuhl Seminar Proceedings, 2006.

[2] E. Cambouropoulos, M. Crochemore, C. Iliopoulos, L. Mouchard, "Algorithms for computing approximate repetitions in musical sequences,"Journal of Computer Mathematics, 79, 11, 1135-1148, 2002.

[3] A.T. Castelo, W. Martins and G.R. Gao,"TROLL-tandem repeat occurrence locator,"Bioinformatics, 18, 634 - 636, 2002.

[4] M. Crochemore, "String matching and periods," Bulletin of the European Association for Theoretical Computer Science, 39, 149-153, 1989.

[5] B. Langmead, C. Trapnel, M. Pop, SL. Salzberg, "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome," Genome Biol, 10, R25, 2009.

[6] X. Liu and L. Wang, "Finding the Region of Pseudo-Periodic Tandem Repeats in Biological Sequences," Algorithms for Molecular Biology, Vol.1, No.1, pp.2, 2006.

[7] R. Kolpakov, G. Bana, G. Kucherov, "mreps: Efficient and flexible detection of tandem repeats in DNA," Nucleic Acids Res, 31:3672 - 3678, 2003.

[8] J.S. Sim, C.S. Iliopoulos, K. Park, W.F. Smyth, "Approximate periods of strings," Theoretical Computer Science, 262, 557-568, 2001.

[9] D. Gussfield, "Algorithms on Strings, Trees, and Sequences," Cambridge University Press, 1997.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Introduction to Algorithms," 3nd ed. MIT Press, 2001.

[11] H. Hyyrö, K. Narisawa, S. Inenaga, "Dynamic Edit Distance Table under a General Weighted Cost Function," SOFSEM 2010: Theory and Practice of Computer Science, 5901, 515-527, 2010.

[12] S.R. Kim and K. Park, "A dynamic edit distance table," Journal of Discrete Algorithms, 2, 303 - 312, 2004.

정 주 희

e-mail : jngjuhe@gmail.com
2011년 인하대학교 컴퓨터정보공학부(학사)
2011년~현 재 인하대학교 컴퓨터정보공학과 석사과정
관심분야 : String Algorithm, Bioinformatics

김 영 호

e-mail : yhkim8505@gmail.com
2011년 인하대학교 컴퓨터정보공학부(학사)
2011년~현 재 인하대학교 컴퓨터정보공학과 석사과정
관심분야 : String Algorithm, Parallel Algorithm

나 중 채

e-mail : jcna@sejong.ac.kr
1998년 서울대학교 컴퓨터공학과(학사)
2000년 서울대학교 컴퓨터공학과(석사)
2005년 서울대학교 컴퓨터공학과(박사)
2006년 Department of Computer Science, University of Helsinki, Postdoctoral researcher
2007년 건국대학교 u-science 기반 신기술 융합사업단 연구교수
2008년~현 재 세종대학교 컴퓨터공학과 조교수
관심분야 : Theory of Computation, Algorithm, Bioinformatics

심 정 섭

e-mail : jssim@inha.ac.kr
1995년 서울대학교 컴퓨터공학과(학사)
1997년 서울대학교 컴퓨터공학과(석사)
2002년 서울대학교 컴퓨터공학과(박사)
2002년 3월~2004년 8월 한국전자통신연구원 바이오정보연구팀선임연구원
2004년 9월~현 재 인하대학교 컴퓨터정보공학부 부교수
관심분야 : Theory of Computation, Algorithm, Bioinformatics