대규모 염색체들을 위한 개선된 Probe 선택 알고리즘

권영대*, 박경욱**, 임형석**
*전남대학교 소프트웨어공학 협동과정
**전남대학교 전산학과
e-mail:vy2020@alex.chonnam.ac.kr

An Improved Probe Selection Algorithm for Large Genomes

Young-Dae Kwon*, Kyoung-Wook Park**, Hyeong-Seok Lim**
*Interdisciplinary Program of Software, Chonnam National University
**Dept. Computer Science, Chonnam National University

요 약

유전자 칩의 정확성은 각 유전자들의 식별자로 활용되는 probe들에 의해 결정된다. 일반적으로 칩을 구성하는 probe들은 반응 오류를 예측하기 위해 이중구조와 녹는점과 같은 요소들을 고려한다. 또한 다른 유전자들과의 교차반응을 최소화하기 위해 각 probe들의 specificity도 고려되어야 한다. probe가 specificity를 보장하는지 검증하는 것은 전체 유전자들을 탐색해야 하므로 대규모 인간염색체에 대해서는 많은 시간이 소요된다.

본 논문에서는 specificity를 만족하는 probe들을 선택하는 효율적인 알고리즘을 제시한다. 제시한 알고리즘은 해시 테이블을 활용하여 probe가 specificity를 만족하지 못하게 하는 유전자 시퀀스들만을 탐색하여 비교한다. 제시한 알고리즘이 기존 알고리즘보다 효율적임을 실험결과를 통해 보인다.

1. 서론

유전자 칩(DNA chip)은 작은 공간의 유리 또는 금속표면에 probe라 불리는 짧은 길이의 DNA 시퀀스들을 고밀도로 붙여 놓은 것으로 각 probe들은 각 유전자들을 식별할 수 있는 지문 역할을 한다[1]. 유전자 칩의 정확도를 높이기 위해서는 반응 오류(hybridization error)를 최소화하는 probe들을 선택해야한다. 일반적으로 반응 오류를 예측하기 위해 probe의 이중구조(secondary structure)와 녹는점(melting temperature)과 같은 요소들을 고려한다[2]. 또한 다른 유전자들과의 교차 반응(cross hybridization)을 최소화하기 위해 각 probe들은 specificity를 보장해야한다.

probe의 이중구조나 녹는점에 대한 검증은 선형시간에 수행되지만 specificity를 검증하기 위해서는 전체 염색체를 탐색해야 한다. 염색체의 길이가 N이고 probe의 길이가 m일 때 $O(Nm^2)$의 복잡도를 지니므로 인간 염색체와 같은 대규모의 염색체에 대해서는 specificity 검증에 많은 시간이 소요된다. 이러한 계산량을 줄이기 위해 OligoArray, Oligo-Selection등과 같은 probe 선택 도구들은 휴리스틱 알고리즘을 이용한다[3]. 그러나 이러한 기법들은 probe의 specificity를 보장하지 못하므로 선택한 probe들이 교차 반응을 일으킬 수 있다. 최근 [4]에서는 보다 적은 계산으로 specificity를 보장하는 probe들을 선택하는 알고리즘(FindProbe)을 제시하였다.

본 논문에서는 specificity를 보장하는 probe들을 선택하는 효율적인 알고리즘을 제시한다. 제시한 알고리즘은 해시 테이블을 활용하여 probe가 specificity를 만족하지 못하게 하는 유전자 시퀀스들만을 탐색하여 비교함으로써 적은 계산으로 specificity를 검증할 수 있다. 제시한 알고리즘은 시뮬레이션을 통해 S.pombe, S.cerevisiae, Neurospora crassa 같은 대규모 염색체들의 probe들을 선택하는데 기존의 FindProbe보다 효율적임을 보인다.

논문의 구성은 다음과 같다. 2장에서는 기존 연구

들에 대해 소개하고 3장에서 specificity를 보장하는 효율적인 probe 선택 알고리즘을 제시한다. 4장에서는 제시한 알고리즘을 분석하고 시뮬레이션 결과를 보이며 5장에서 결론을 맺는다.

2. 관련연구

일반적으로 유전자 칩에 삽입되는 probe들의 선택은 homogeneity, sensitivity, specificity 방법을 이용한다[5]. 유전자 칩에 삽입될 probe를 찾는 문제는 다음과 같이 정의할 수 있다.

Probe 선택 문제 유전자들의 집합 $G = \{g_1, g_2, ..., g_n\}$이라 하자. 길이 m인 probe를 선택하는 문제는 모든 유전자 g_i에 대해 homogeneity, sensitivity, specificity를 만족하는 probe들의 집합 $P = \{p_1, p_2, ..., p_n\}$을 찾는 것이다. 이때 $p_i \in g_i$이고 p_i의 시퀀스 길이 $|p_i| = m$이다.

2.1 Homogeneity 필터링

Homogeneity 필터링은 정해진 범위 내의 녹는점(melting temperature)을 지닌 probe를 후보로 선택하는 것이다. 이 방법은 좋은 probe들은 그들이 반응할 유전자의 일부분과 비슷한 온도에서 반응한다는 점을 이용한다. 후보 probe들 중 지나치게 낮거나 높은 온도를 갖는 probe들은 그들이 반응해야 할 목적 probe들과 정확하게 반응하지 못하거나 다른 probe와 반응 오류를 일으킬 수 있으므로 제외시킨다[2].

2.2 Sensitivity 필터링

Sensitivity 필터링은 2중 구조를 갖는 probe들을 제거하는 것이다. 이 방법은 probe의 3′ 끝에서 길이 x인 세그먼트를 선택하고 이 세그먼트가 probe 내에서 길이 y만큼의 연속된 보수 세그먼트 형태이면 이 probe는 2중 구조를 지니는 것이므로 제외시킨다.

2.3 Specificity 필터링

Specificity 필터링은 교차반응을 일으키는 probe를 제외시키는 것이다. 유전자들의 집합 $G = \{g_1, g_2 ..., g_n\}$이라 하고 유전자 g_i에 있는 길이가 m인 후보 probe p가 specificity를 만족하는지를 검사하기 위해서는 유전자집합 $g' = G - \{g_i\}$에 있는 길이 m인 모든 서브스트링 q에 대해 해밍 distance $H(p, q) \geq w$인지를 계산해야 한다. 여기서 w는 교차반응을 일으킬 수 있는 임계값(threshold)이다. 만약 임의의 q에 대해 $H(p, q) < w$이면 교차반응이 일어날 수 있으므로 제외시킨다.

Brute Force방법은 전체 염색체의 길이가 N일 때 복잡도 $O(Nm^2)$으로 대규모의 염색체에 대해서는 많은 시간이 소요된다. 이를 위해 specificity를 보장하지는 못하지만 가능한 교차반응을 최소화하는 probe들을 선택하는 휴리스틱 알고리즘들이 제안되었다.

Li와 Stormo[6]는 suffix array와 myersgrep을 이용한 휴리스틱 알고리즘을 제안하였으며 [3]에서는 specificity 검증을 위해 BLAST 데이터베이스를 활용하고 이중구조를 검증하기 위해 Mfold를 이용하였다. 또한 [7]에서는 공통 최장 스트링(Longest Common Substring)을 이용한 휴리스틱 알고리즘을 제안하였다. 이러한 알고리즘들은 적은 시간이 소요되지만 specificity를 보장하지 못하므로 선택된 probe들은 교차반응을 일으킬 수 있다. 최근 Sung와 Lee[4]는 비둘기집의 원리를 이용하여 후보 probe와 교차반응을 일으킬 수 있는 시퀀스들을 탐색하여 검증함으로써 Brute Force 알고리즘에 비해 낮은 복잡도를 지닌 알고리즘을 제시하였다.

3. 제안한 specificity 필터링 알고리즘

본 논문에서는 probe 선택 문제를 해결하는 효율적인 알고리즘을 제안한다. 먼저 주어진 염색체 G의 모든 유전자 g_i에 대해 g_i의 길이 m인 서브스트링을 후보 probe로 하고 다음의 세 단계를 거치면서 최종 probe를 선택한다.

(단계 1) homogeneity를 만족하지 못하는 probe들을 후보 probe에서 제외시킨다.
(단계 2) sensitivity를 만족시키지 못하는 probe들을 후보 probe에서 제외시킨다.
(단계 3) 나머지 후보 probe들 중에서 specificity를 만족하는 probe를 선택한다.

단계 1과 2의 homogeneity 필터링과 sensitivity 필터링은 앞의 2.1절과 2.2절의 방법들을 이용하여 후보 probe들을 필터링 한다. 그리고 이들 필터링을 거친 후보 probe들 중에서 specificity를 만족하는 probe를 단계 3에서 검사하고 이를 만족하면 최종 probe로 선택한다.

두 개의 스트링이 w 미만의 mismatch를 지닌 경우 다음 [보조정리1]과 같은 성질을 만족한다.

보조정리 1[8]. 두 개의 길이 m인 문자열 $p[1,...,m]$와 $q[1,...,m]$의 $H(p,q)<w$이면 $H(p',q')=0$인 p와 q의 부 문자열 p'와 q'가 다음과 같이 존재한다. 여기서 $k=\lfloor m/(w-1)\rfloor$이다.

(a) $p'=p[i,...,i+k-1]$, $q'=q[i,...,i+k-1]$.

(b) $p'=p[i, i+\frac{m}{k}, ..., i+(k-1)\frac{m}{k}]$,
$q'=q[i, i+\frac{m}{k}, ..., i+(k-1)\frac{m}{k}]$.

교차반응을 일으킬 수 있는 임계값 w가 주어졌을 때 보조정리 1에 의해 길이 m인 probe p가 나쁜 probe가 되려면, 다시 말해 p를 포함하지 않는 나머지 유전자들의 부 시퀀스를 q라 할 때 $H(p,q)<w$가 되려면, 길이 k인 m/k개의 p'에 대해 $H(p',q')=0$을 만족하는 q'들의 위치를 검색하여 $H(p,q)<w$인지를 검사하면 된다. 이때 모든 q'에 대해 $H(p,q)\geq w$이면 p는 specificity를 만족한다.

본 논문에서는 probe p의 부 시퀀스 p'를 보조정리 1(b)로 설정하고 $H(p',q')=0$인 q'를 빠르게 탐색하기 위하여 해시 테이블을 활용한다. 유전자들은 A, G, T, C의 4가지로 구성되어 있으므로 이들을 두 자리의 이진수로 대응시킬 수 있다(A→00, G→01, T→11, C→10). 먼저 크기가 4^k인 해시 테이블 HT를 생성하고 각각의 슬롯에는 연결리스트(linked list)를 저장한다. 그리고 염색체의 전체 시퀀스에 대해 (Algorithm 1)과 같이 해시 테이블을 생성한다. 이때 사용할 해시 함수는 다음과 같다.

$$h(s,m,k)=s[1]\times 4^{k-1}+s[1+\frac{m}{k}]\times 4^{k-2}+...+s[1+(k-1)\frac{m}{k}]\times 4^0$$

예를 들어 $k=4$이고 $s=\mathrm{AGTC}$이면 HT의 "00011110" 슬롯에는 s를 포함하는 모든 유전자들의 번호와 시퀀스 위치에 대한 정보가 연결리스트로 저장된다. 따라서 i번째 유전자의 j번째 위치에 있는 probe $g_i[j,...,j+m-1]$의 specificity를 검증하기 위해 (Algorithm 2)와 같이 $\delta=h(g_i[j+r-1], m, k)$를 계산하고 HT_δ의 연결리스트에 저장된 위치 정보를 이용하여 임계값 w보다 적은 해밍 distance를 갖을 가능성이 있는 시퀀스만을 찾아 비교한다.$(1\leq r\leq m/k)$.

특히 염색체는 두 개의 유전자 g_x와 g_y가 연속된

Algorithm 1. 해시테이블 생성

Input : 염색체 G, probe 크기 m, threshold w
Output : 해시테이블 HT
Procedure ConstructHashTable(G, m, w)
  $k=\lfloor m/(w-1)\rfloor$
  각 주소마다 list를 지닌 4^k크기의 해시테이블 HT생성
  for $i:=1$ **to** n **do**
    for $j:=1$ **to** $|g_i|-m-(m/k)$ **do**
      $\delta=h(g_i[j],m,k)$
      HT_δ의 list에 (i,j)추가
    end for
  end for
  return HT

Algorithm 2. specificity 검증

Input : 염색체 G, i번째 유전자 j번째 sequence $g_i[j]$, probe 크기 m, threshold w, 해시테이블 HT
Output : $g_i[j]$부터 bad probe의 개수 $skip$
Procedure CheckSpecificity(G, $g_i[j]$, m, w, HT)
  $skip:=0$
  for $r:=1$ **to** m/k **do**
    $\delta=h(g_i[j+r-1], m, k)$
    HT_δ list의 모든 항목 (p,q)에 대해 $(p\neq i)$
      $q':=q-r$
      $hd:=$HammingDistance($g_i[j]$, $g_p[q']$, m)
      if $hd<w$ **then**
        $g_i[j]$와 $g_p[q']$를 bad probe로 설정하고
        $skip$, j, q'를 1씩 증가시키면서
        $hd\geq w$일때까지 $g_i[j]$와 $g_p[q']$를 비교하며
        bad probe 판별
      end if
  end for
  return $skip$

유사한 시퀀스를 지니는 경우가 많다. 따라서 $g_x[i]$와 $g_y[j]$의 해밍 distance가 w보다 적으면 이들 둘을 나쁜 probe로 설정하고 i와 j를 1씩 증가시키면서 해밍 distance를 구해 이들이 연속적으로 유사한 시퀀스인지에 대해 비교함으로써 연속적으로 나쁜 probe가 발생할 때 보다 빠르게 처리되도록 하였다.

(Algorithm 3)에서는 염색체 G의 각각의 유전자에 대해 specificity를 만족하는 probe들의 집합 P를 선택하는 전체 과정을 나타낸다.

4. 분석 및 평가

염색체의 전체 시퀀스의 길이가 N인 경우 HT의 각 연결리스트들에는 평균 $\frac{1}{4^k}N$개의 위치 정보가 저장된다. 하나의 probe가 specificity를 만족하는지

Algorithm 3. specificity 만족하는 probe 선택

Input : 염색체 G, probe 크기 m, threshold w, 해시테이블 HT
Output : probe 집합 $P = \{p_1, p_2, \ldots, p_n\}$
Procedure SelectSpecificityProbe(G, m, w, HT)

```
for i := 1 to n do
  for j := 1 to |g_i| − m + 1 do
    if g_i[j] = bad_probe
      continue
    skip := CheckSpecificity(G, g_i[j], m, w, HT)
    if skip = 0 then
      p_i := g_i[j, ..., j+m]
      break
    else
      j := j + skip − 1
    end if
  end for
end for
return P
```

검증하기 위해서는 길이 k인 m/k개의 부 시퀀스들에 대한 위치 정보를 이용하여 비교하므로 $O(\frac{m}{k}\frac{1}{4^k}Nm)$의 시간 복잡도를 갖는다. 따라서 전체 n개의 유전자들에 대해 각각 하나의 probe를 선택하므로 복잡도는 $O(\frac{m^2}{k}\frac{1}{4^k}Nn)$이다.

제안된 알고리즘은 C++로 구현하여 펜티엄4 2.8GHz에서 [표 1]과 같이 세 종류의 염색체들에 대해 테스트하였다. 기존의 FindProbe와 같이 probe의 길이 $m = 50$, 임계값 $w = 15$로 설정하였다. 테스트 결과 [그림 1]과 같이 specificity 필터링에 FindProbe보다 18-50%정도 적은 시간이 소요되었다. 특히 염색체의 크기가 커질수록 보다 더 적은 시간이 소요되므로 대규모의 염색체에서 보다 효율적임을 보여준다.

5. 결론

본 논문에서는 specificity를 보장하는 probe들을 선택하는 효율적인 알고리즘을 제시하였다. 제시된 알고리즘은 해시 테이블을 활용하여 probe가 specificity를 만족하지 못하게 하는 유전자 시퀀스들만을 탐색하여 비교함으로써 적은 계산으로 specificity를 검증할 수 있다. 또한 기존의 소프트웨어들에 의해 선택된 probe들이 specificity를 만족하는지 검증하는데 활용될 수 있다.

표 1. 테스트 데이터

Genome Name	Number Of Genes	Length(bps)
S. pombe	4997	7.1×10^6
S. cerevisiae	6343	8.9×10^6
Neuro. crassa	10895	1.5×10^7

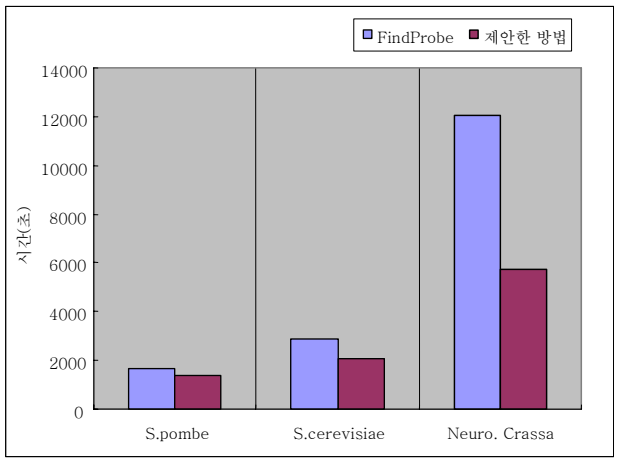

그림 1. 시뮬레이션 결과

참고문헌

[1] Gerhold D./ Rushmore T. and Caskey C. T. DNA chips: promising toys have become powerful tools. In Trends in biochemical sciences, pages 168-173, 1999.

[2] Keller GH., Keller GH Manak MM. DNA Probes Second Edition, chapter Section 1: Molecular hybridization technology, Stockton Press, pp. 1-9, 1993.

[3] Rouillard J. M., Herbert C. J. and Zuker M. Oligoarray:Genome-scale oligonucleotide design for microarrays. Bioinformatics(Applications Note), 18:486-487, 2002.

[4] Wing-Kin Sung and Wah-Heng Lee. Fast and Accurate Probe Selection Algorithm for Large Genomes. proceedings of the Computational Systems Bioinformatics(CSB), 2003.

[5] Lockhart D.J.,Dong H.C., Follettie M.T., Gallo M. V.,Chee M.S.,Mittmann M.,Wang C.,Kobayashi M., Horton H.and Brown E.L. Expression monitoring by hybridization to highdensity oligonucleotide arrays.Nature Biotechnology,14:1675-1680, 1996.

[6] Li F. and Stormo G, Selection of optimal DNA oligos for gene expression analysis, Bioinformatics, vol. 17, pp. 1067-1076, 2001.

[7] Lipson D., Webb P., and Yakhini Z. Designing Specific Oligonucleotide Probes for the Entire S.cerevisiae Transcriptome. WABI, LNCS 2452: 491-505, 2002.

[8] P. A. Pevzner and M. S. Waterman, Multiple Filteration and Approximate Pattern Matching, Algorithmica, vol. 13, pp. 135-154, 1995.