Improved Algorithm for Haplotype Block Partitioning : Application to Human Chromosome 21

Kyoung Rak Na, Sang Jun Kim,, Sung Kwon Kim*

Department of Computer Science & Engineering, Chung-Ang University, Korea

***To whom correspondence should be addressed. E-mail: skkim@cau.ac.kr**

Abstract

Research of basis technology to construct the human haplotype map is one of active areas in SNP post-genomics research. Identification of haplotype block structure from haplotype data is key step in the haplotype map project. Several algorithms have been proposed for the block identification, including the greedy algorithm, and the dynamic programming based algorithm. This paper analyzed block partitioning method of several algorithm which has been proposed in recent years. HapBlock and HaploBlockFinder are programs used in our experiment.

Introduction

Human Genome Project 이후의 프로젝트로 인간 genome의 질병 유전자를 지도로 만드는 *haplotype map* (HapMap)을 만들기 위해 Haplotype mapping 연구 [1,2,3]가 진행되고 있다. Haplotype mapping은 개개인마다 달라질 수 있는 질병에 대한 소인과 약물에 대한 반응 등을 고려한 특화된 치료, 약 처방, 예방을 가능케 하기 위해서 haplotype과 phenotype간의 연관성을 찾아내는 과정이다.

이러한 연관성을 밝히는데 가장 큰 장애는 인간 유전체가 가지고 있는 SNP 개수의 대량성이며, 이로 인해 특정 질병과 연관된 돌연변이 근처의 marker를 전체 genome에 대해서 mapping하여 질병과 연관된 유전자를 찾기 위해서는 많은 시간과 비용이 들게 된다. 그러나 haplotype은 블록으로 나누어 분석이 될 수 있다는 사실이 최근의 연구들 [2~10]에 의해 밝혀지면서, 적은 시간과 비용이 드는 mapping의 길이 열리게 되었다.

Haplotype을 블록들로 나누어 분석해 보면, population의 90%를 차지하는 haplotype의 개수는 4-5개 정도라는 것을 알 수 있다. 따라서 common haplotype을 찾아내고 각각을 구분해 주는 대표 SNP을 2-3개 정도 찾아서 그것으로 common haplotype을 표현할 수 있다. 이러한 제한된 블록의 다양성을 가지는 염색체에서 대표 SNP들을 이용하면 전체 패턴을 대부분 재생할 수 있게 된다.

haplotype 블록 분석을 위한 블록 분할에서는 Greedy algorithm [2,12]과 Dynamic

본 연구는 한국 과학재단 목적기초연구 (R01-2003-000-11573-0)지원으로 수행되었음.

programming- based algorithm [7,8]이 이용되어, 염색체 데이터들로부터 common haplotype과 대표 SNP의 근사해와 최적해를 얻어내는 방법들이 제시되어 있다.

본 연구는 haplotype 블록 추론 알고리즘의 개발에 앞서 현재까지의 접근 방식에 대해 실험과 분석 수행하였다. 이를 통하여 새로운 알고리즘을 개발하는 과정에서 기존 연구와 비교 하였을 때 전산학적인 최적해 뿐만 아니라 생물학적으로도 좋은 결과를 보이는 방법의 모색과 성능의 향상 가능성을 살펴보았다.

Methods

Chromosome 21의 24,047개 SNP [2]들을 가진 20개 sample로 구성된 데이터 셋을 입력으로 하여 현재까지 개발된 알고리즘 [2,7,8,12]과 프로그램 [7,8,12]을 이용하여 블록 분할을 수행하였다.

블록 분할을 위해 사용된 프로그램은 Greedy algorithm으로 구현된 HaploBlock Finder (Kun Zhang et al., 2003 [12])와 Dynamic programming-based algorithm으로 구현된 HapBlock (Zhang et al., 2003 [7,8])을 이용하였다.

실험의 결과로 나온 데이터를 통해 common haplotype들과 대표 SNP들을 비교분석하였으며, 알고리즘 각각에 대해 수행결과를 비교분석 하였다.

자료분석 프로그램

Notation

B : SNP들의 수로 정의된 블럭
L(B) : 블록의 길이(SNP들의 수)
f(B) : common haplotype들을 구분하기 위해 필요한 SNP들의 최소수

Patil et al.

Greedy algorithm을 사용하여 L(B)/f(B)의 비를 계산하여 가장 큰 값을 가지는 구간을 블록으로 정의한다. (그림 1) Patil et al. [2]의 80% 적용범위에 대해 블록 분할된 데이터는 Perlegen (www.perlegen.com)에서 제공된 데이터를 사용하여 비교 분석하였다.

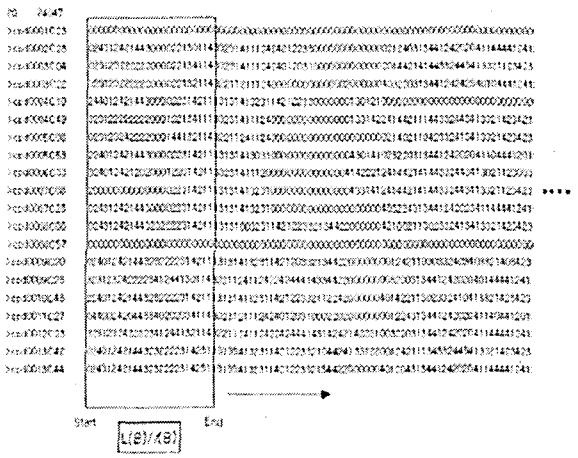

그림 1. Patil et al. 알고리즘 모식도

HaploBlockFinder

그림 2의 모식도에서 나타낸 것과 같이 블록분할을 목적하는 지역안의 모든 가능한 블록을 평가하여 가장 큰 구간을 블록으로 정한다.

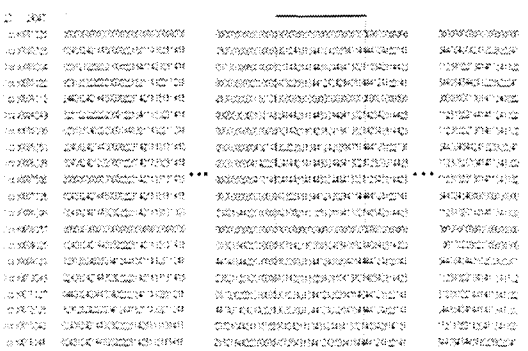

그림 2 HaploBlockFinder 알고리즘 모식도(1)

다음으로 그림 3에서와 같이 정해진 블록의 양쪽 지역에 대해 모든 SNP들이 블록으로 할당 받을 때까지 재귀적으로 반복하여 블록 분할을 수행한다.

그림 3 HaploBlockFinder 알고리즘 모식도(2)

HapBlock

분할된 블록이 $B_1,...,B_l$와 같을 때 최적 블록 분할은 대표 SNP들의 수를 최소화하는 블록분할 ($f(B_1)+...+f(B_l)$)로 정의한다. SNP들은 다음의 식 [7,8]을 따라 계산되어 최적 블록분할이 수행된다. (그림 4)

S_j는 첫 j SNP들($r_1,r_2,...,r_j$)의 최적 블록분할을 위한 대표 SNP들의 수로 정의한다. (이때 $S_0=0$.) Dynamic programming 이론을 적용하여 재귀를 이용해 최소 대표 SNP을 가지는 블록분할을 계산할 수 있다.

$$S_j=\min\{S_{i-1}+f(r_i,...,r_j),\ \text{if}\ 1\le i\le j\ \text{and}\ \text{block}(r_i,...,r_j)=1\}.$$

C_j는 S_j개 대표 SNP들을 필요로 하는 블록분할에서 블록이 최소의 블록으로 분할되는 경우일 때 블록의 수라 정의한다.

$$C_j=\min\{C_{i-1}+1,\ \text{if}\ 1\le i\le j\ \text{and}\ \text{block}(r_i,...,r_j)=1\ \text{and}\ S_j=S_{i-1}+f(r_i,...,r_j)\}.$$

각각의 식을 되풀이하여 최종적으로 최소 대표 SNP들을 가지는 최소 블록의 수인 C_n이 최적해를 가지는 경우를 계산하게 된다.

이때 블록의 정의는 Patil et al. [2]의 Chromosome coverage로 지정하여 블록 분할을 수행하였으며, 블록 분할을 통해 얻어낸 common haplotype으로부터 α%의 chromosome을 재구성 할 수 있는 값인 적용범위는 각각 α = 70, 80, 90으로 설정하였다.

실험에 쓰인 24,047개 SNP [2]들은 minor 대립유전자의 빈도가 0.1이상인 common SNP들이며, common haplotype은 각 블록에서 두 번 이상 나타난 haplotype으로 정의한다.

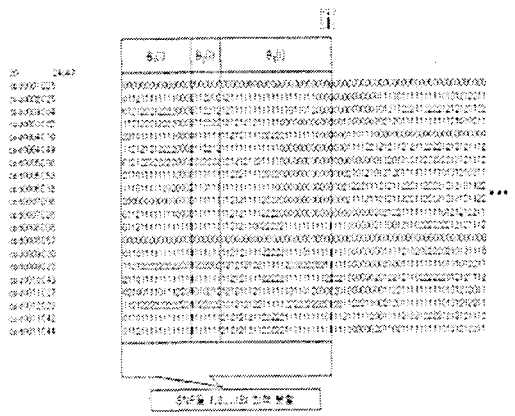

그림 4. HapBlock 알고리즘 모식도

자료분석 방법

블록으로 분할된 각각의 데이터로부터 common haplotype과 각 pattern과 특성을 비교하였다. Greedy 방식과 Dynamic programming 알고리즘을 사용한 각각의 프로그램에서 얻어진 근사해와 최적해는 어떤 차이를 보이는지, 또한 같은 알고리즘을 사용한 방법에서는 블록을 정의하는 방식에 따라 결과 값의 차이가 얼마나 달라지는 것인지 비교분석 하였다.

전체 데이터에서 common haplotype의 분포를 알아보기 위해 각 블록에서 가장 많은 분포를 가지는 상위 3개와 4개의 common haplotype이 몇 %를 차지하는지 분석하였다.

데이터를 블록으로 분할하였을 때 나타나는 common haplotype의 패턴으로부터

population에서의 haplotype의 다양성을 비교할 수 있다.
또한 common haplotype의 정보를 알아내어 이를 다시 표현하기 위해 필요한 대표 SNP들의 개수를 구하여 각 방법의 효율을 비교하였다.

평가 데이터

실험에 사용한 데이터는 34Mb에 걸친 4개 Contig (NT-002836, NT- 001035, NT-003545, NT-002835)로 구성된 Chromosome 21이며, rodent-human somatic cell hybrid 기술을 이용해 얻은 35,989개 SNP들 중 minor 염기가 하나만 있는 site를 가지는 11,603개의 singletons과 339개의 셋 또는 그 이상의 allele를 가지는 SNP들은 제외되었고, minor allele가 한번 이상 나타난 24,047 개의 SNP [2]들을 Perlegen (www.perlegen.com)으로부터 제공받아 실험에 사용하였다. 이 데이터는 African, Asian, Caucasian로 구성된 20개의 sample로 구성되어 있으며, 각 인종의 구분은 나타나 있지 않다.

Results

실험에 사용한 프로그램에서 블록을 정의하는 criteria는 Patil et al. [2]과 같은 방법으로 지정하였다. Patil te al. [2]의 결과는 다른 방법과 비교를 위해 80% 적용범위의 데이터를 표 1에 보였다. 그러나 70, 90%의 적용범위에 따르는 결과를 제공하지 않아 결과 분석에 포함시키지 않았다.
80% 적용 범위 일 때 Dynamic Programming 방법은 전체 블록을 2575개로 분할하여 가장 최적의 분할을 보였으며, Greedy 알고리즘을 사용한 HaploBlockFinder [12]의 결과는 블록을 2999개로 나누었고, 이는 4135개로 나누어진 Patil et al. [2]의 방법 보다 월등한 결과를 보였다. 알고리즘을 구성하는 방법에 따라 성능 향상의 가능성을 볼 수 있었으며, 좋은 결과를 보인 HapBlock에서도 보다 좋은 결과를 얻을 수 있는 방법을 구성할 수 있을 것이다.

Method	Common SNPs/block	No. of Blocks	Average size/block, kb	All blocks, %	Common SNPs, %
HapBlock	>10	742	24.27	28.8	75.5
	3-10	909	7.30	35.3	19.5
	<3	924	0.73	35.9	5.0
	Total	2575	12.58	100.0	100.0
Patil et al.	>10	589	23.90	14.2	56.8
	3-10	1408	8.52	34.1	30.7
	<3	2138	2.96	51.7	12.4
	Total	4135	7.83	100.0	100.0
HaploBlock Finder	>10	856	18.83	28.5	67.0
	3-10	1120	6.17	37.4	28.7
	<3	1023	1	34.1	4.3
	Total	2999	8.02	100.0	100.0

표 1. 80%의 적용범위를 가지는 각 방법들에서 haplotype 블록의 특성

표 2의 결과를 보면 70%일 때에는 Dynamic programming방법과 Greedy방법의 격차가 많이 줄어들었으며, 그림 7에서 나타낸 것과 같이 3개 이상의 분포에 있어서는 Greedy 방법의 경우가 더 적은 수의 블록을 생성하는 것을 볼 수 있다. 그러나 블록의 평균 길이 분포를 보면 Dynamic programming방법의 경우 평균적으로 더 긴 길이를 가지며, 최적 블록 분할은 Dynamic programming에 의해 얻어지게 된다.

Method	Common SNPs/block	No. of Blocks	Average size/block, kb	All blocks, %	Common SNPs, %
HapBlock	>10	650	30.78	27.3	79.3
	3-10	703	7.47	29.5	14.6
	<3	1028	0.60	43.2	6.1
	Total	2381	13.61	100.0	100.0
HaploBlock Finder	>10	866	18.51	35.8	74.0
	3-10	938	6.15	38.8	23.4
	<3	613	1	25.4	2.6
	Total	2417	9.95	100.0	100.0

표 2. 70%의 적용범위를 가지는 각 방법들에서 haplotype 블록의 특성

다음으로, 표 3에서는 90%일 때의 결과를 보여주며 평균 길이에 있어서는 많은 차이가 나지 않으나 블록의 개수에 있어서는 확연히 차이가 나는 것을 볼 수 있다.

Method	Common SNPs/block	No. of Blocks	Average size/block, kb	All blocks, %	Common SNPs, %
HapBlock	>10	669	17.41	18.7	52.9
	3-10	1724	6.28	48.3	40.0
	<3	1180	0.84	33.0	7.1
	Total	3573	9.07	100.0	100.0
HaploBlock Finder	>10	780	18.51	17.5	60.0
	3-10	1151	6.15	25.8	29.5
	<3	2526	1	56.7	10.5
	Total	4457	5.40	100.0	100.0

표 3. 90%의 적용범위를 가지는 각 방법들에서 haplotype 블록의 특성

이러한 차이는 아래의 그림 5에서 나타낸 것과 같이 3보다 작은 common SNP들이 잘 분석되지 않는데서 발생한다.

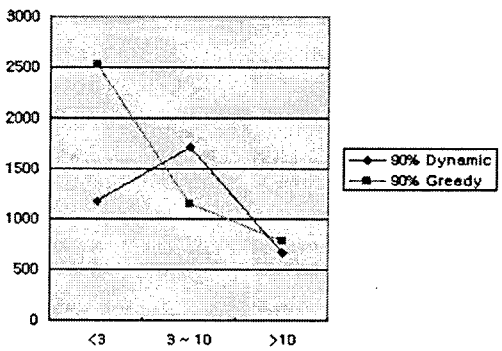

그림 5. 90% 적용범위에서 Common haplotyle의 길이별 분포된 블록의 개수

그림 5, 6에서와 같이 부분적으로 Greedy 방법이 좋은 결과를 보이기도 하나, 그림 8에서도 보이는 바와 같이 적용 범위가 커질수록 격차가 벌어지는 것을 볼 수 있으며, Dynamic programming을 이용했을 때가 전반적으로 최적의 블록 분할을 수행한다는 사실을 확인할 수 있었다. 물론 chromosome coverage를 통해 얻은 이러한 블록은 해당 블록이 생성된 생물학적인 근거는 포함되지 않았으며 [6,7], 생물학적인 의미의 최적해를 보장하는 것은 아니다.

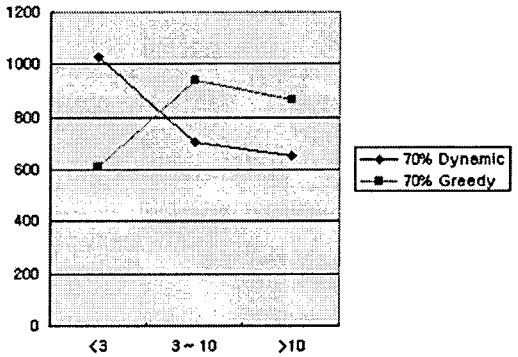

그림 6. 70% 적용범위에서 Common haplotyle의 길이별 분포된 블록의 개수

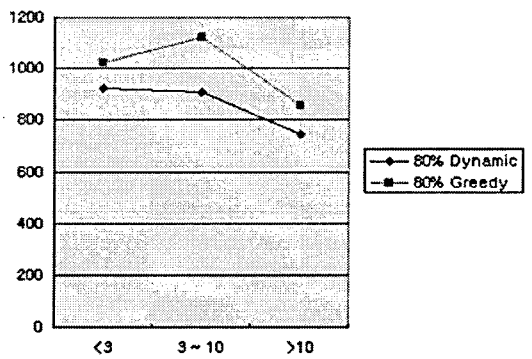

그림 7. 80% 적용범위에서 Common haplotyle의 길이별 분포된 블록의 개수

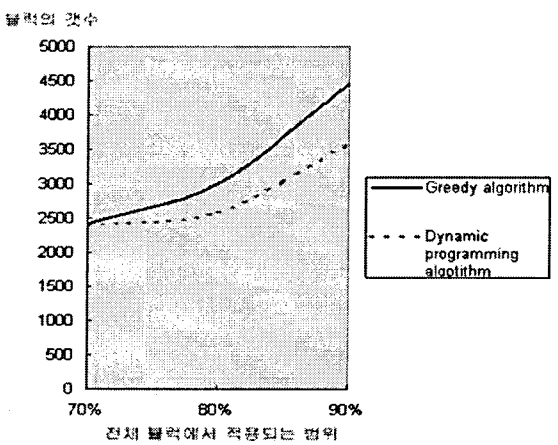

그림 8. 적용범위에 따라 각 알고리즘 별로 분할되는 블록 개수의 차이

전체 블록에서 나타나는 common haplotype의 분포를 알아보기 위해 20개의 sample에서 가장 긴 길이의 common haplotype의 패턴을 각각 3개와 4개씩 계산하였다.

이는 각 블록에서 나타나는 일부 common haplotype의 패턴이 전체의 몇 %를 차지하는지 나타내게 된다. 표 4와 표 5에서와 결과와 같이 population에서 3-5 개의 common haplotype으로 대부분의 패턴을 나타낼 수 있음을 알 수 있다. 따라서 common haplotype과 대표 SNP을 이용하여 전체 패턴을 재생하는 방법은 genotyping에 드는 시간과 비용을 줄이며 매우 효과적인 방법이 될 수 있다.

Method	Chromosome Coverage, α%	common haplotype patterns/block (3개)			Ambigous haplotypes /block	Coverage
Patil et al.	80%	9.6	4.2	2.1	–	79.5%
HapBlock	70%	9.2	3.2	1.3	4.3	86.0%
	80%	8.7	3.3	1.5	4.3	88.3%
	90%	8.0	3.8	1.8	4.6	87.3%
HaploBlock Finder	70%	9.2	3.2	2.3	1.1	62.5%
	80%	8.7	3.3	2.4	1.4	69.9%
	90%	8.0	3.8	2.4	2.1	87.7%

표 4. 각 Block안에서의 Common Haplotype의 pattern별 분포와 범위 (상위 3개)

Method	Chromosome Coverage, α%	common haplotype patterns/block (4개)				Ambigous haplotypes /block	Coverage
HapBlock	70%	9.2	3.2	1.3	0.9	4.3	93.1%
	80%	8.7	3.3	1.5	0.7	4.3	91.7%
	90%	8.0	3.8	1.8	1.0	4.6	93.6%
HaploBlock Finder	70%	9.2	3.2	2.3	1.1	1.1	68.3%
	80%	8.7	3.3	2.4	1.1	1.4	75.7%
	90%	8.0	3.8	2.4	0.7	2.1	91.7%

표 5. 각 Block안에서의 Common Haplotype의 pattern별 분포와 범위 (상위 4개)

Stephens et al. [13]은 313개의 유전자에 대한 haplotype 분석에서 82%의 haplotype이 모든 ethnic group에서 관찰되며, 8%만 인종에 따라 특정하다는 결과를 보였다.

common haplotype에 포함되지 않으며 인종에 따라 특정한 8%의 부분들이 유전자 발현부위에 존재하는 변이일 경우, 기존 방법에서는 모호성을 띄는 haplotype으로 판단되어 분석에서 제외될 수 있으며 이는 연관연구에 있어 power를 낮추는 결과를 보일 수 있다. 따라서 블록이 생성된 생물학적인 근거를 포함하여 블록 분할을 수행하는 방법과 uncommon haplotype과 같이 기존의 블록 분할 방법으로는 잘 분석되지 않는 부분들에 대해서도 좋은 결과를 얻어 낼 수 있는 블록 분할 방법을 구성한다면 전산학적으로 최적해이면서, 생물학적으로 의미 있는 결과를 산출하는 결과를 얻게 될 것이라 예상한다.

Discussion

Greedy 알고리즘과 Dynamic Programming Algorithm을 이용한 블록 분할 프로그램을 통하여 블록분할을 실험한 결과 Dynamic Programming 알고리즘을 이용한 HapBlock [7]의 경우가 보다 좋은 결과를 보여주는 것을 확인할 수 있었다. 그러나 수행시간에서는 Greedy 알고리즘을 이용한 HaploBlockFinder [12]가 10배 이상 빠르게 결과를 출력하였다. 따라서 최적해를 보장하며 동시에 수행시간을 최소한으로 줄이는 알고리즘의 성능 향상이 필요하다.

대부분의 SNP은 유전자 사이에 위치하여 생체기능의 변이를 동반하지 않는 intergenetic SNP이며, 아미노산 생성에 영향을 주어 개인의 약물 반응의 다양성의 원인이 되는 coding SNP과 perigenic SNP이 유전자 변이가 인체에서 어떤 영향을 미치는가를 평가하는 상관성 연구에서 중요한 의미를 가진다. 블록 분할에 있어서 이러한 생물학적인 의미가 반영되도록 성능의 향상을 모색해야 할 것이다.

본 연구는 전산학적인 의미의 최적해가 반드시 생물학적으로도 가장 좋은 결과를 주

는 것만은 아니라는 가정에 기반한다. 또한 실험에서 얻어진 결과를 바탕으로 궁극적으로 전산학과 생물학적인 의미를 모두 만족시키며 목적함수에 대해 최적해를 보장하는 블록 분할 알고리즘을 개발하고 이를 구현하는데 목표를 두고 있다.

References

[1] J. Couzin, "New Mapping Project Splits the Community ", SCIENCE, vol. 296, pp. 1391-1393, May 2002.

[2] N. Patil, et al, "Blocks of Limited Haplotype Diversity Revealed by High-Resolution Scanning of Human Chromosome 21", SCIENCE, vol. 294, pp. 1719-1722, November 2001.

[3] S. B. Gabriel, et al, "The Structure of Haplotype Blocks in the Human Genome", SICENCE, vol. 296, pp. 2225-2229, June 2002.

[4] H. I. Avi-itzhak, "Selection of Minimum Subsets of Single Nucleotide Polymorphisms to Capture Haplotype Block Diversity", Pacific Symposium on Biocomputing pp. 466-477, 2003.

[5] M. Koivisto, "An MDL Method for Finding Haplotype Block and For Estimating The Strength of Haplotype Block Boundaries ", Pacific Symposium on Biocomputing 2003, pp. 502-513, 2003.

[6] V. Bafna, "Haplotype and Informative SNP Selection Algorithms : Don't Block Out Information", RECOMB 2003

[7] K. Zhang, "A Dynamic Programming Algorithm for Haplotype Blocks partitioning", PNAS, vol. 99. no. 11, pp 7335-7339, May 2002.

[8] K. Zhang, "Dynamic Programming Algorithms for Haplotype Block Partitioning : Applications to Human Chromosome 21 Haplotype Data", ACM RECOMB, pp. 332-340, 2003.

[9] D. Posada and C. Wiuf, "Simulating haplotype blocks in the human genome", Bioinformatics, vol. 19. no. 2, pp. 289-290, 2003.

[10] J. Zhang, "HapScope: a software system for automated and visual analysis of functionally annotated hapotypes", Nucleic Acids Research, vol. 30. no. 23, pp 5213-5221, 2002.

[11] M. Nothnagel, "Simulation of LD block-structured SNP haplotype data and its use for the analysis of case-control data by supervised learning methods", Am J Hum Genet 71 (Suppl.)(4): A2363. 2002.

[12] Kun Zhang, Li Jin, "HaploBlockFinder: Haplotype Block Analyses", Bioinformatics, vol. 19, pp. 1300-1301, 2003.

[13] J.C. Stephens, et al. "Haplotype variation and linkage disequilibrium in 313 human genes", Science vol. 293, pp. 489-493, 2001.