# COMPARISON OF ALGORITHMS FOR THE HAPLOTYPE ASSEMBLY PROBLEM

**Ömer Nebil YAVEROĞLU**
**Middle East Technical University**
**Computer Engineering Department**
**June 2009**

# PROBLEM DEFINITION

- Haplotype Assembly Problem
- Finding the whole haplotype sequence from a collection of fragments which :
  - come from two complementary haplotypes
  - represents small portions of the haplotype
  - Can be erroneous
- Haplotype Assembly Problem is proven to be NP-Hard.

# AIM OF THE PROJECT

- Comparing the three well-known approximation algorithms for the haplotype assembly problem.
  - HASH – Bansal et al. [1]
  - Fast HARE – Panconesi et al. [2]
  - HapCUT – Bansal et al. [3]
- Understanding the proof of NP-Hardness of the problem.

# DATASETS

- Full haplotype sequence of human is not available as a reference haplotype so no ready to use dataset for the purposes of the project exist.
- Bansal et al. uses HuRef reference haplotype data for the testing purposes of the algorithm.
- Panconesi et al. generate a syntethic dataset of haplotypes as similar as it can be to real haplotypes.

# NP-Hardness of the problem

- The short proof from Panconesi et al.[2] says:

Haplotype Assembly Problem

↓

Minimum Fragment Removal Problem
(Bipartite Graph Generation)

↓

Minimum Element Removal Problem

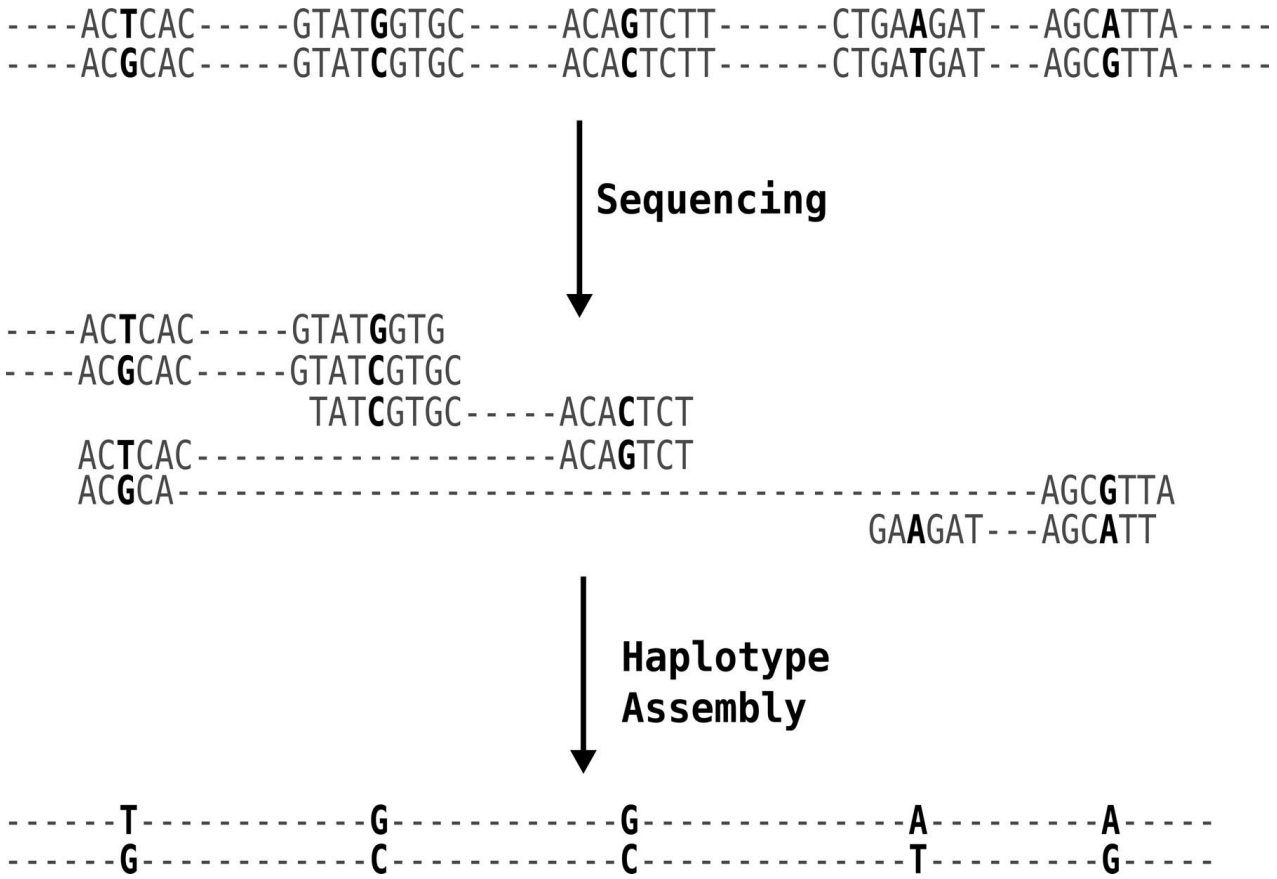
↓

Hypercube Segmentation Problem
(Given to be NP-Hard [4])

# Methods

- All of the methods try to find the separation of the fragments into two sets one representing the fragments coming from the first haplotype and the other coming from the second haplotype.
- Instead of considering the whole haplotype, the data is simplified by just considering single nucleotide polymorphism locations (SNP's) where the genetic variations occur.
- For this purpose all three methods start by constructing a SNP fragment matrix. The rows of this matrix corresponds to fragments and the columns corresponds to SNP locations on the haplotypes.

# METHODS


----ACTCAC-----GTATGGTGC-----ACAGTCTT------CTGAAGAT---AGCATTA-----
----ACGCAC-----GTATCGTGC-----ACACTCTT------CTGATGAT---AGCGTTA-----
Sequencing
----ACTCAC-----GTATGGTG
----ACGCAC-----GTATCGTGC
TATCGTGC-----ACACTCT
ACTCAC-----------ACAGTCT
ACGCA-----------AGCGTTA
GAAGAT---AGCATT
Haplotype
Assembly
------T------------G------------G-------------A---------A-----
------G------------C------------C-------------T---------G-----

# HASH

- Uses Markov Chain Monte Carlo Methods
- Converts the fragment matrix into a graph where…
  - Nodes represent columns of fragment matrix
  - Edges represent the coverage of columns
  - Weights of edges show the number of fragments that covers both columns
- Tries to find the minimum cut of this graph

# HASH

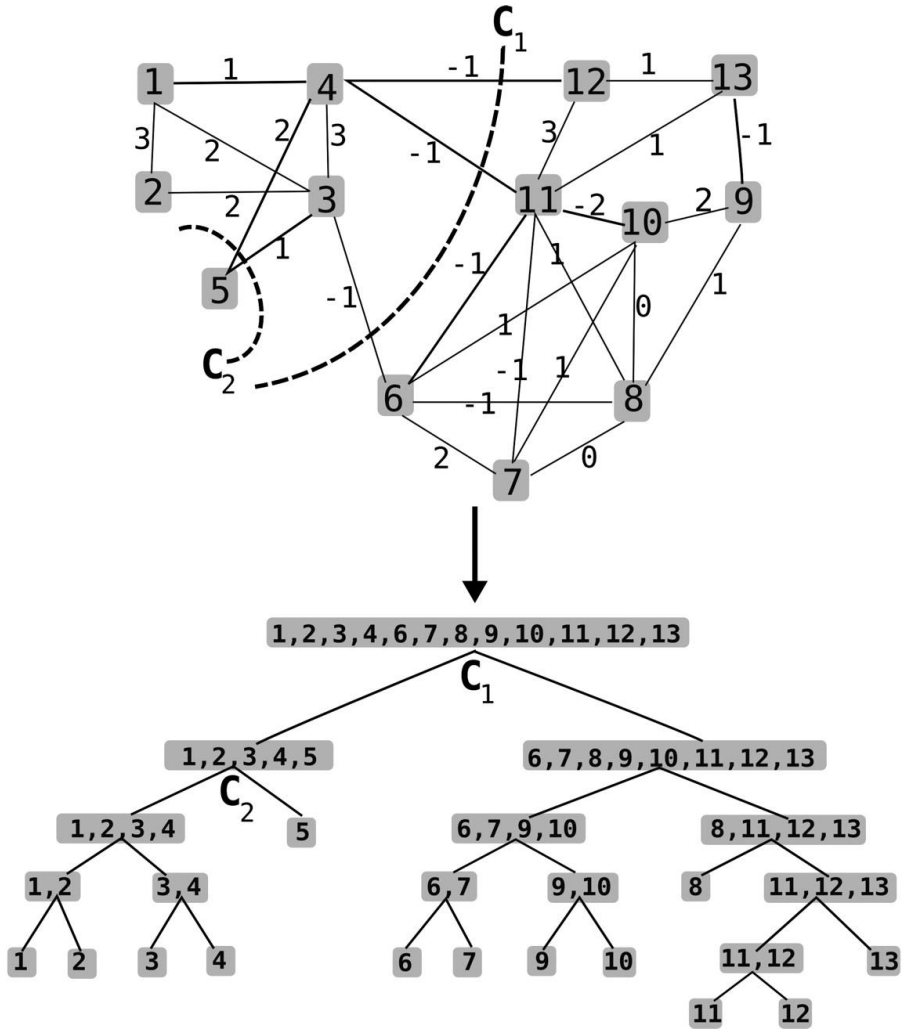
```
0000000001111
1234567890123
```

X

```
00----------
--11---------
0000---------
111----------
--001--------
--0--0-------
---0------11-
---10--------
-----000-----
-----1-0-----
-----001-00--
-------000---
--------11---
----------000-
----------000
--------1---1
```

H

```
0000111111000
1111000000111
```



# FAST HARE

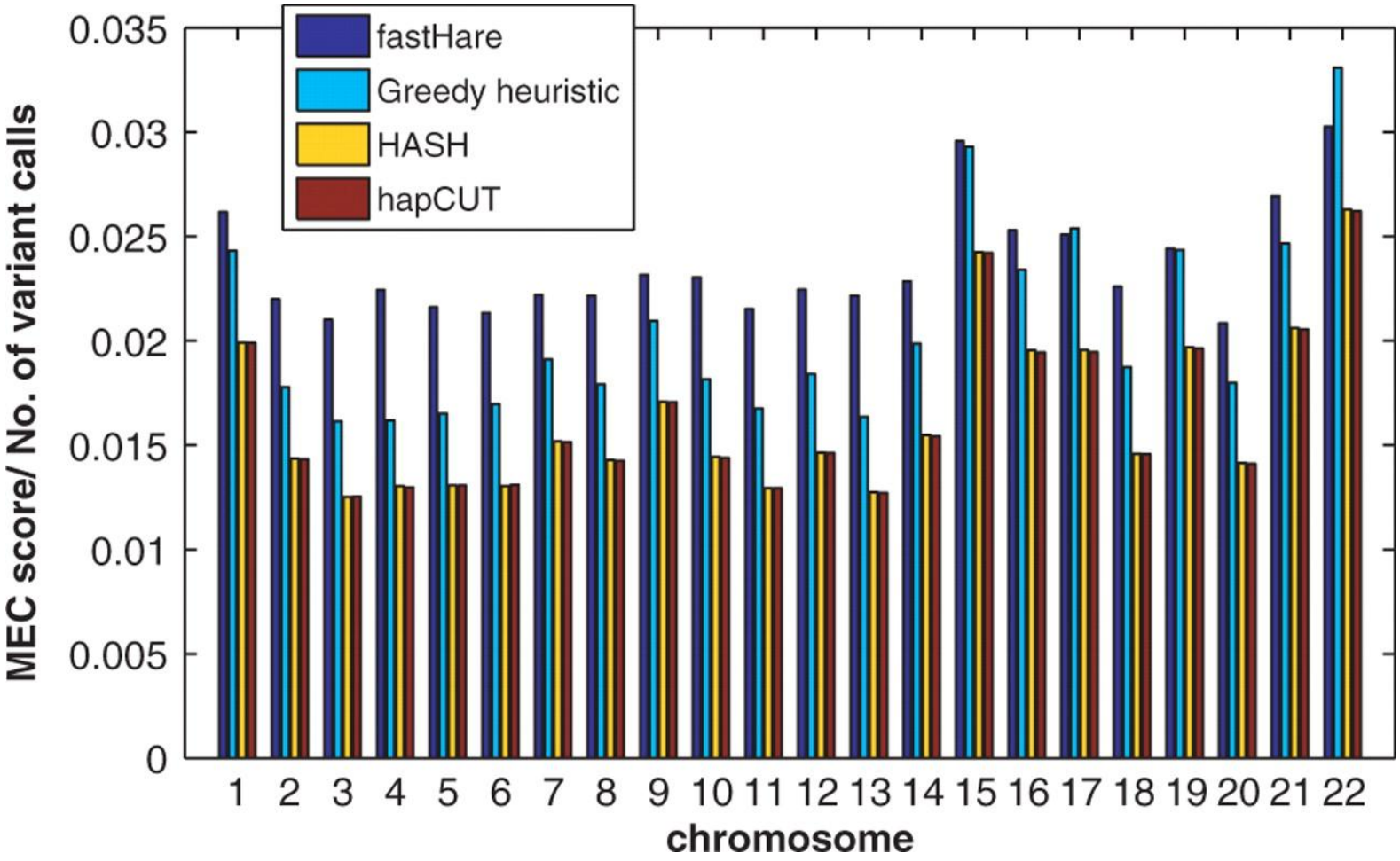
- A simple algorithm which aims to separate the given fragments into two classes according to their in between distances

- The main algorithm is as follows:
  1. Eliminate the columns in which fa ≤ t or fb ≤ t.
  2. Sort the rows according to the ones starting with non-null stretch.
  3. Partition the rows of the matrix constructed according to which haplotype they represent.

# HapCUT

- Similar to HASH expect it looks for maximum cuts instead of minimum cuts using the MEC score as the weights of the edges
- Different scoring mechanisms can also be applied.
- The algorithmic complexity of graph partitioning is reduced.

# Comparison of the algorithms



# Comparison of the algorithms

- HASH ~= HapCUT > FastHARE
- HapCUT works quite faster than HASH.
  ( 30 minutes vs. 10 hours)
- The general trend of trying to find the haplotype directly from the fragments have changed to classifying the fragments according to which haplotype they come from.
- Improvement can be achieved by trying different classification techniques.

# REFERENCES


- [1] Vikas Bansal, Aaron L. Halpern and Nelson Axelrod, “An MCMC algorithm for haplotype assembly from whole-genome sequence data”, Genome Research, vol. 18, pp. 1336 - 1346, 2008.
- [2] Alessandro Panconesi and Mauro Sozio, “Fast Hare: A fast heuristic for single individual SNP Haplotype Reconstruction”, WABI, pp. 266-277, 2004.
- [3] Vikas Bansal and Vineet Bafna, “HAPCUT: an efficient and accurate algorithm for haplotype assembly problem”, ECCB, pp. i153-i159, 2008.
- [4] G. Lancia, “Mathematical Programming Approaches for Computational Biology Problems”,In Modelli e Algorithmi per l’Ottimizzazione di Sistemi Complessi, Agnetis and Di Pillo Eds. Pitagore Editrice, 265-310.