High Performance Computing

ELIXIR-EXCELERATE Train-the-Researcher HPC course

Oswaldo Trelles

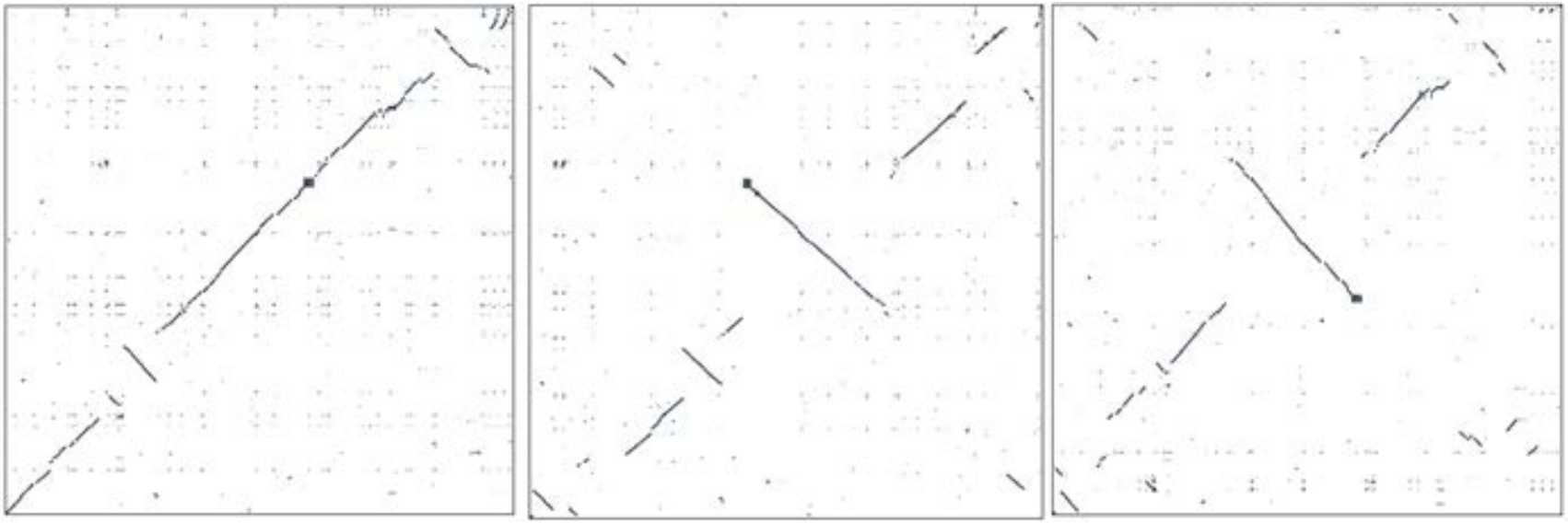
ortrelles@uma.es

European Life Sciences Infrastructure for Biological Information

www.elixir-europe.org

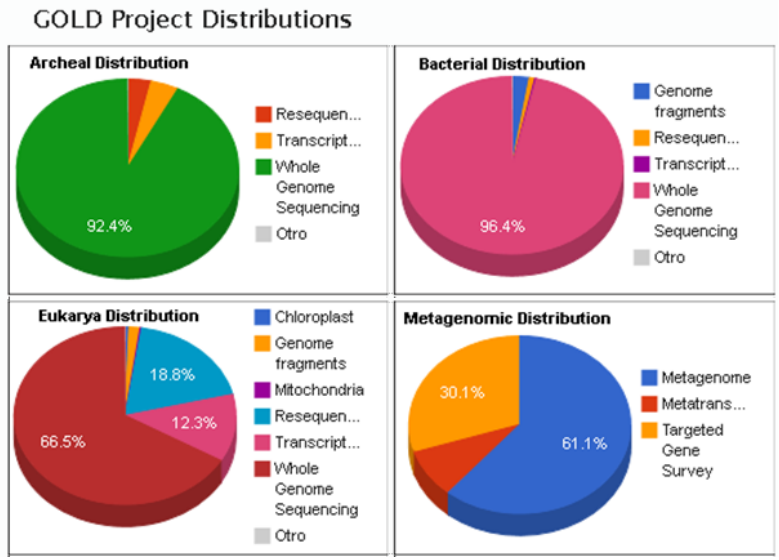
ELIXIR-EXCELERATE is funded by the European Commission within the Research Infrastructures programme of Horizon 2020, grant agreement number 676559.

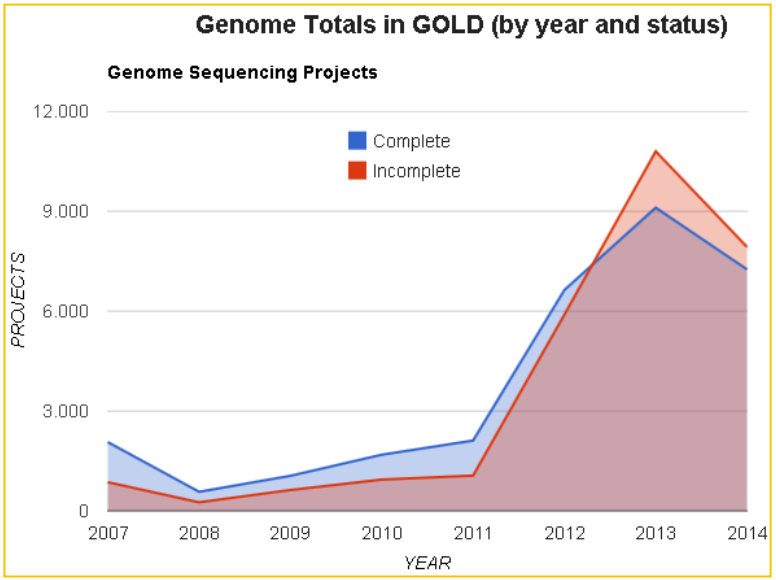
Genome Comparison

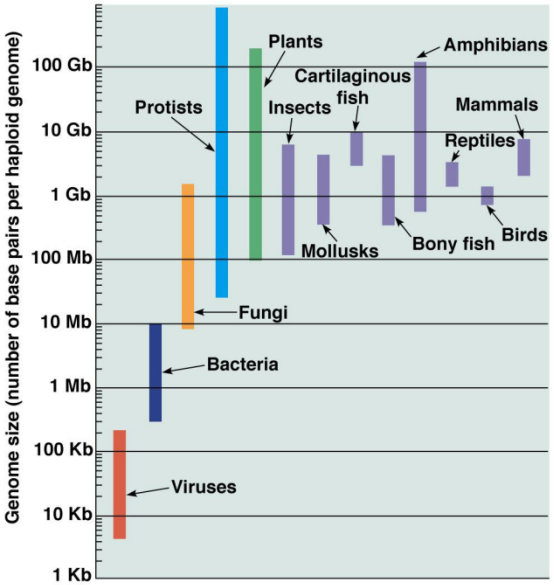


GECKO: Out of core comparison algorithms

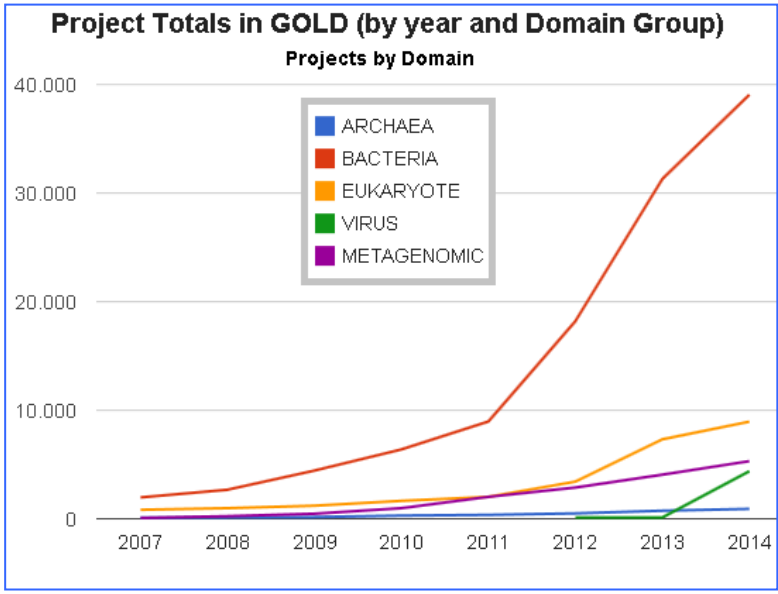
The problem





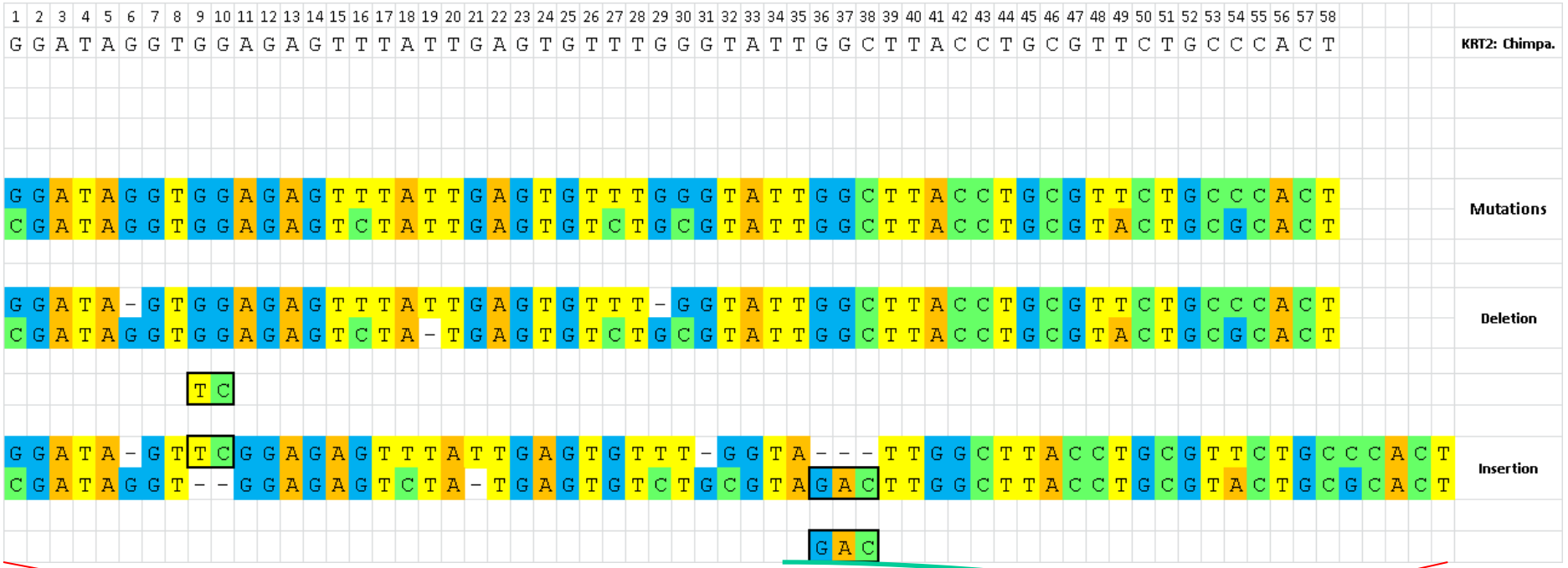


Mammalians	
H. sapiens	Human
M. musculus	Mouse
R. norvegicus	Rat
O. cuniculus	Rabitt
G. gorilla	Gorilla
M. mulata	Macaca
P. trogodytes	Chimpanzee
B. taurus	Cow
S. scrofa	Pig
C. familiaris	Dog



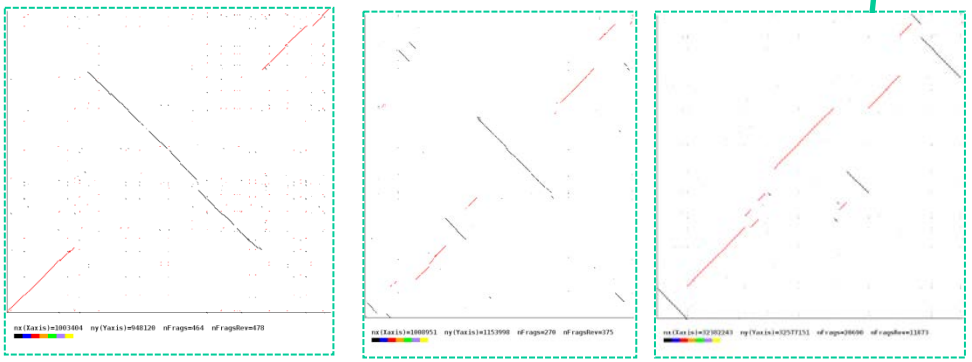
SPAIN

O.Trelles, PhD



Gene-level Evolution Events

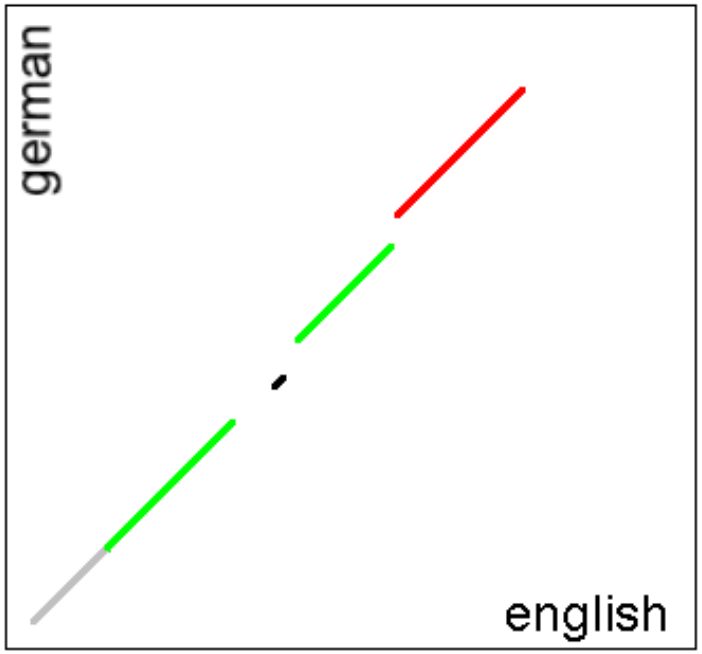
Genome-level

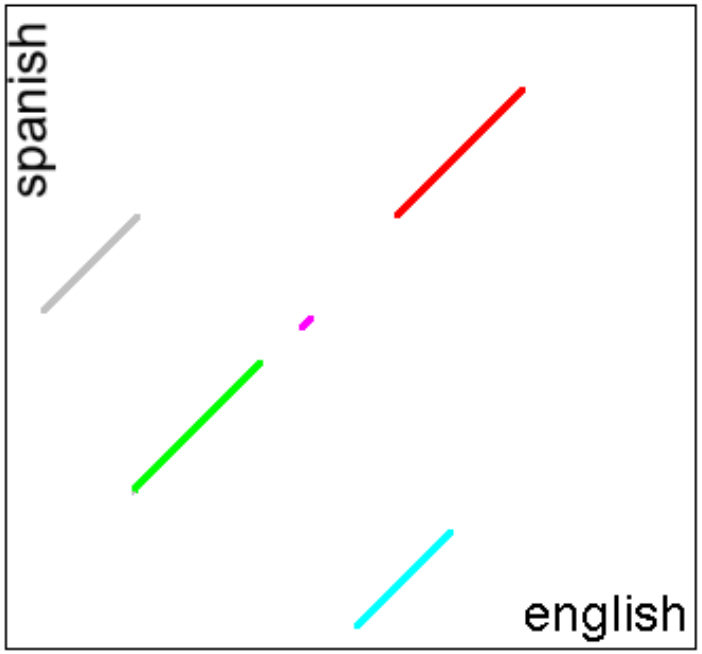


Translocations, inversions, repetitions, duplications, …

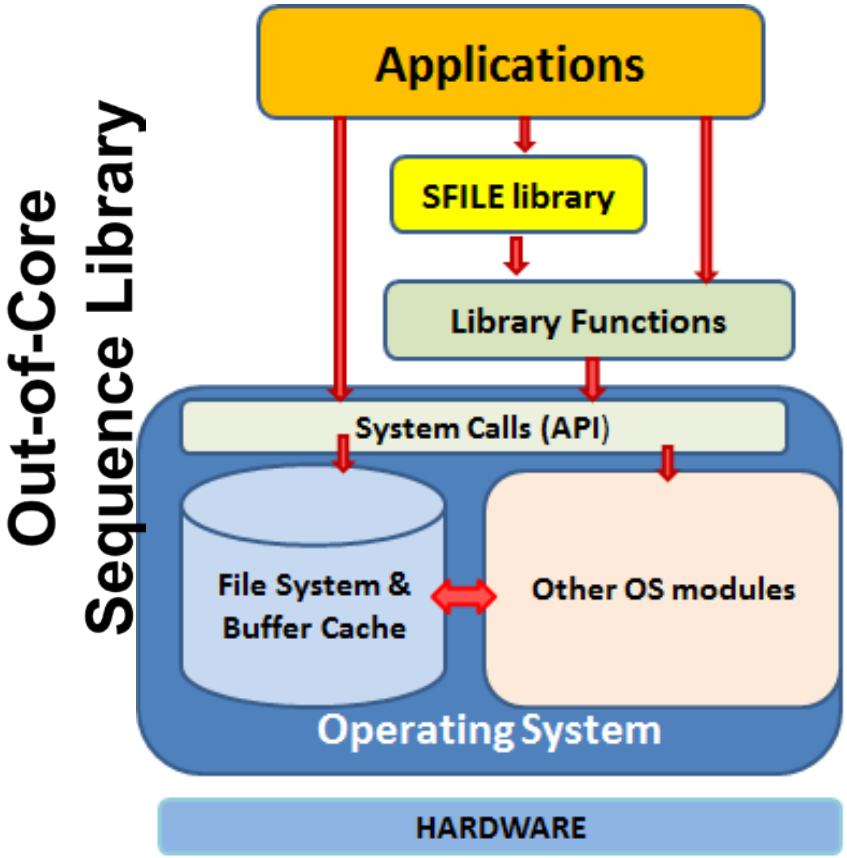
Representation

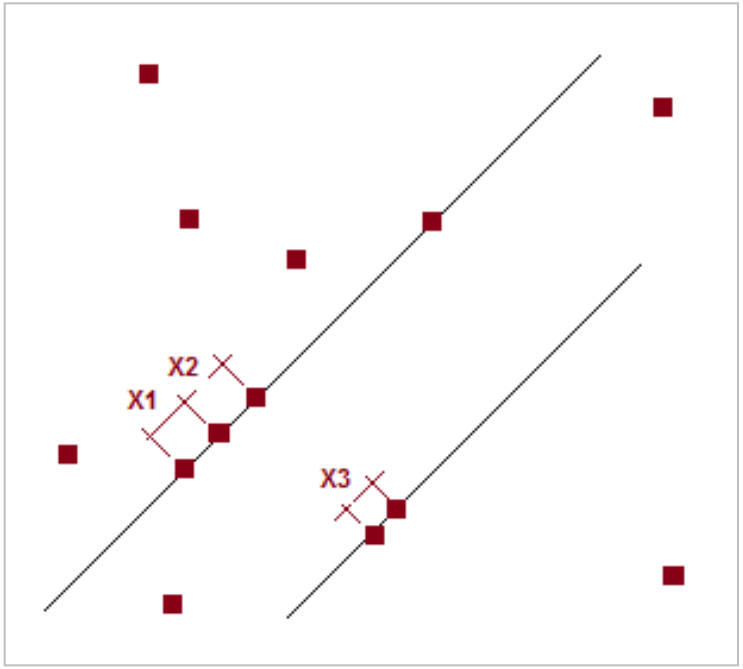
```
THECOMPUTERARCHITECTUREDEPARTMENTAT-THE--UNIVERSITY--OFMALAGA
   ||||||||||||||||.||||          |    |   |||||||||       ||||||
DIECOMPUTERARCHITEKTUREABTEILUNG-AN--DER-UNIVERSITATVONMALAGA
                ||..|||.|||           ||   |||||||||  |     ||||||
ELDEPARTAMENTODEARQUITECTURADECOMPUTADORESDELAUNIVERSIDAD-DEMALAGA
```

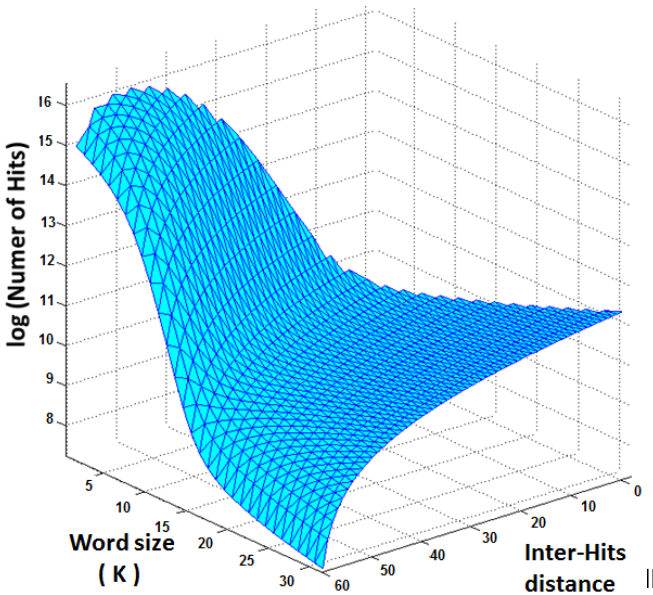




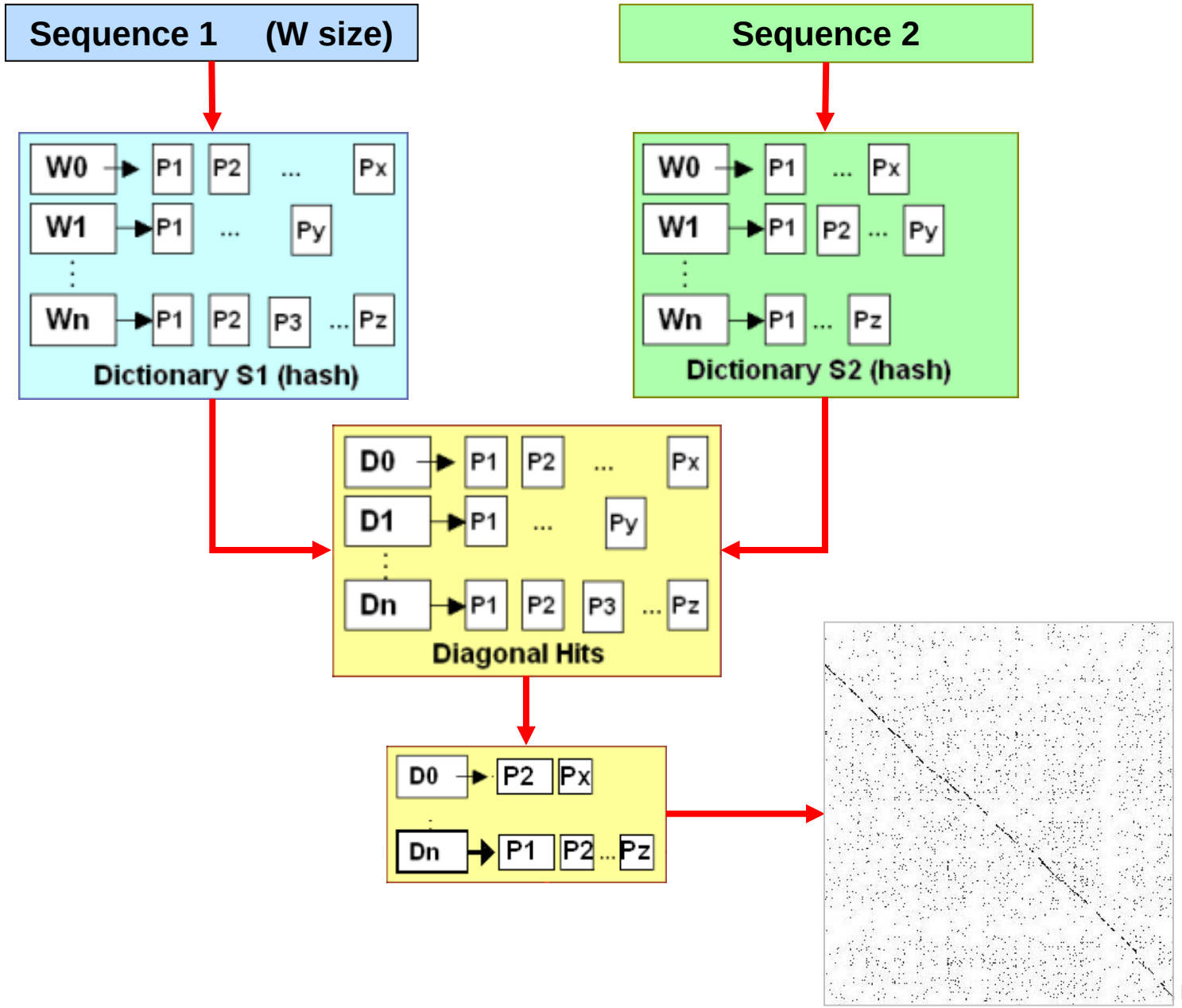
Out-of-Core & Comp-Space-Reduction







Genome Comparison out-of-core: the global idea



K-mers: words of length K

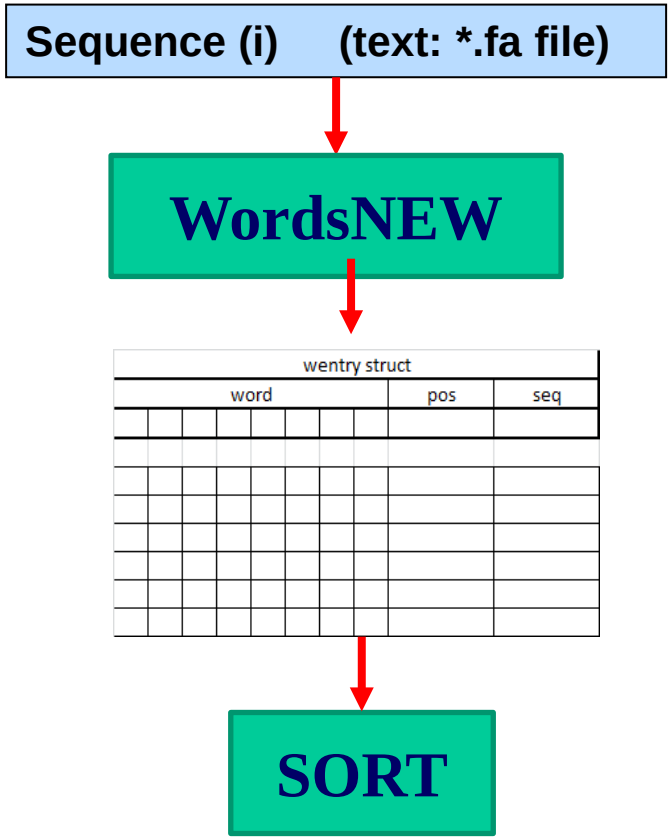
A K-mer is a substring of length *k* in DNA or Protein sequence data

```
actgtagactgactaaggacgggacgggaggactagctaccctattagtt
```

```
actgt
 ctgta
  tgtag
   gtaga
    tagac
     ...
      ...
       cctat
        ctatt
         tatta
          tattag
           attagt
            ttagtt
```

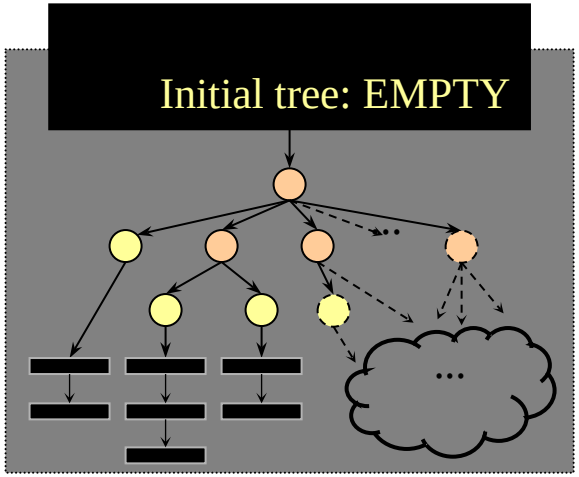
- The exact number of K-mers in a sequence of length L is L-K+1
- The maximum number of different kmers of lenght K is 4^K (DNA) or 20^K (proteins)
- The number of K-mers repetitions (frequency) reduces with the increase of K
- Specific short *k*-mers are called oligomers or "oligos" for short

Building-up the dictionaries




```
Output file format:
typedef struct {
        unsigned char b[8];
} word;
 typedef struct {
        word w;
        unsigned long pos;
        unsigned char seq;
} wentry;
```

Using binary tree instead of sort step

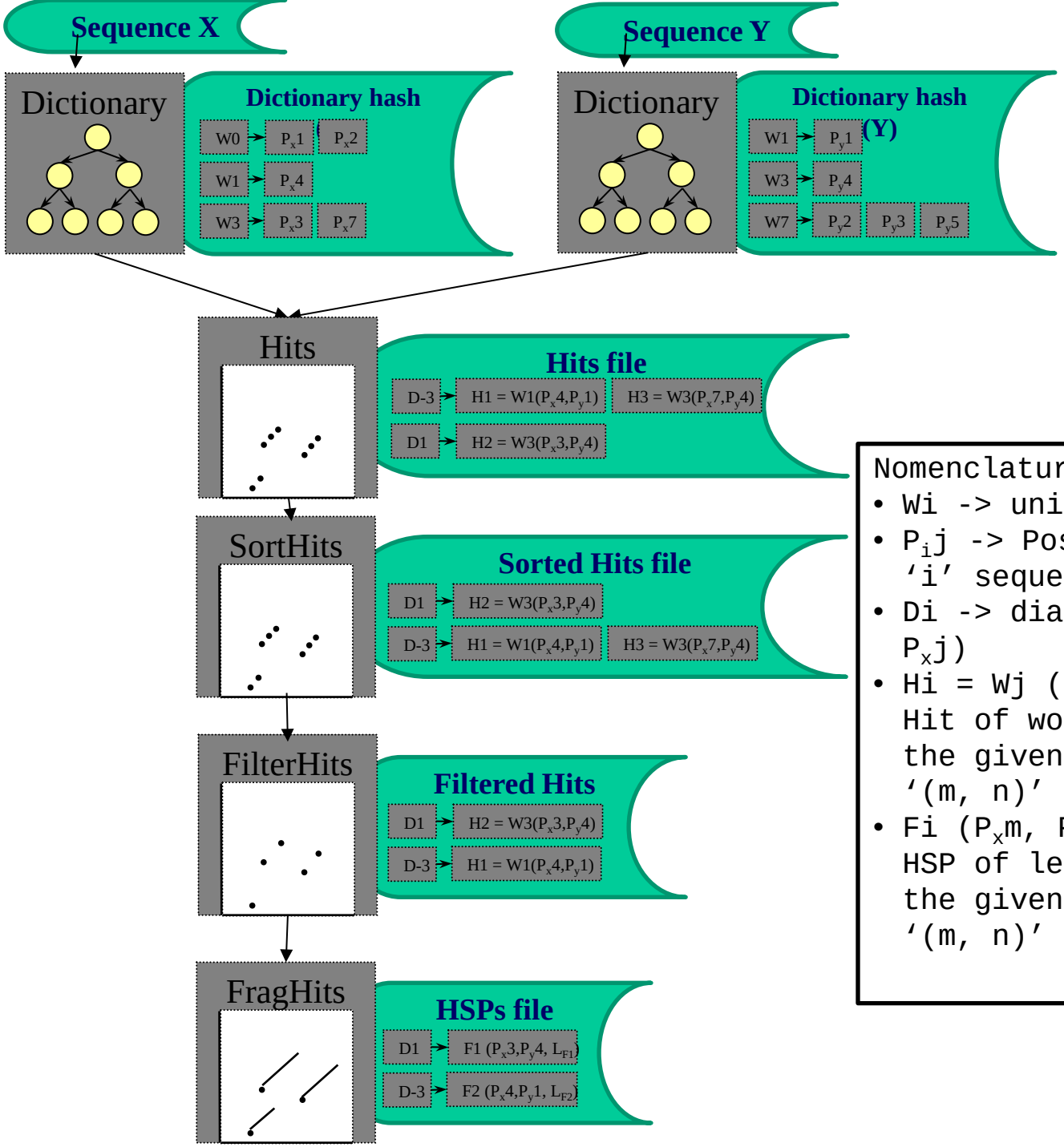



```
Tree data structure
Typedef struct node {
  char key_alue[33]; // word
  struct node *left;
  struct node *right;
  int ocurrences;
  lkst *positions
}
```

```
// Initial tree : EMPTY
seq = readSequence(FILE)
for all kmer
  kmer = nextKmer(seq, wordSize)

  if (kmer < tree.key_value)
        insert(word, tree.left)
  else
    if (kmer > tree.key_value)
        insert(word, tree.right)
  else
        addRepetition
}
```

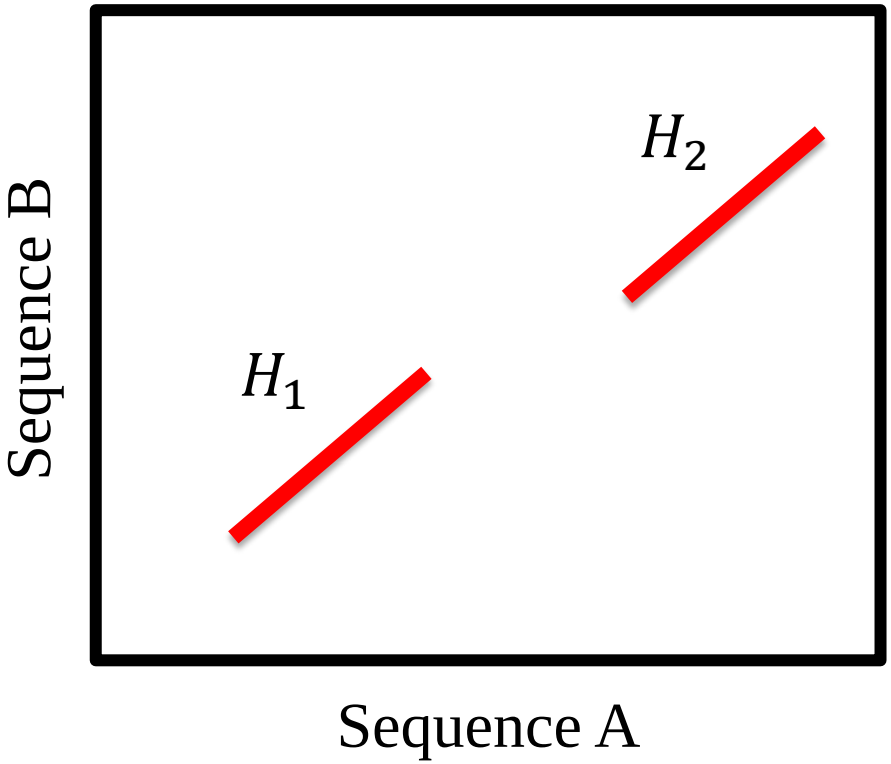
Final workflow



Nomenclature:

- Wi -> unique word ‘i’
- $P_{i}j$ -> Position ‘j’ in ‘i’ sequence
- Di -> diagonal ($P_{y}j$ – $P_{x}j$)
- Hi = Wj ($P_{x}m$, $P_{y}n$) -> Hit of word ‘j’ in the given coordinate ‘(m, n)’
- Fi ($P_{x}m$, $P_{y}n$, L_{Fi}) -> HSP of length ‘L_{Fi}’ at the given coordinate ‘(m, n)’

From fragments to gapped alignments



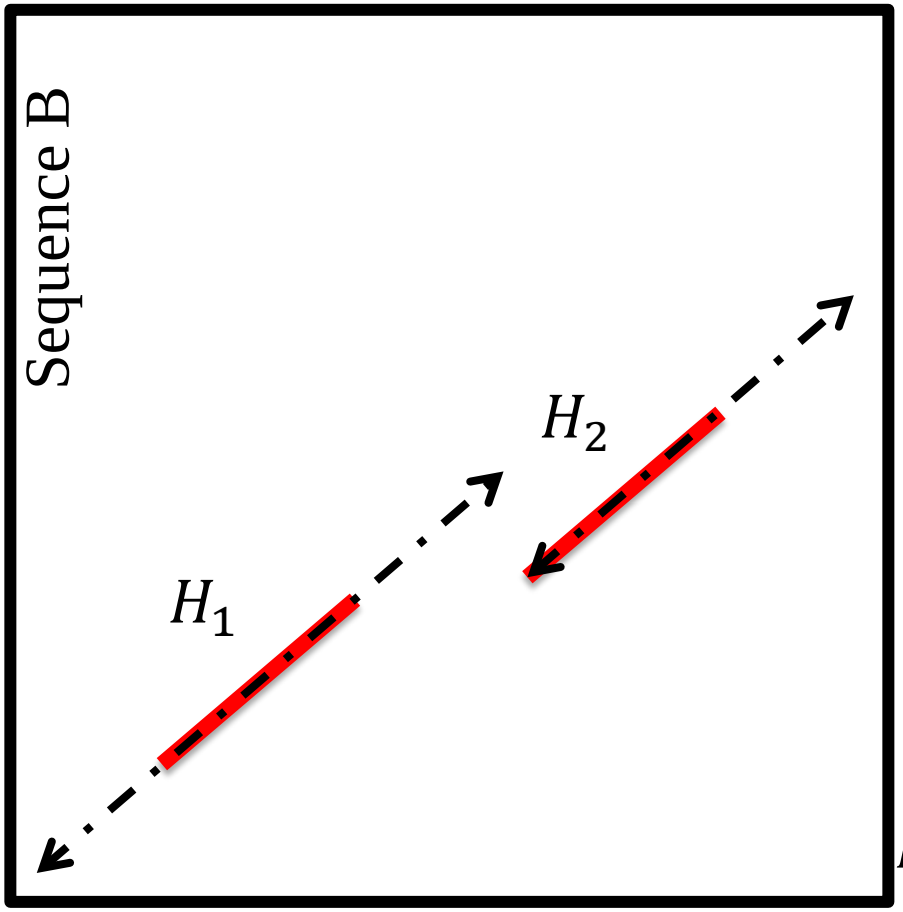
Seed size (k) of 8

```
AAAATACAGTCCCGGCAGGTAGACTTGCGAAACACTTTGATA
|||.|||||||.||||.|.|||...|...|...||||
AAACTACAGTCCGGGCATGAAGAACAACTTGCGAAACAGATA
```

H_1 H_2

Find initial hits by building k-mer dictionaries for both sequences. Shared k-mers are considered hits and will be the seeds for the gapped alignments.

Extending the hits



Extend the initial seeds forward and backwards. The algorithm should maximize a scoring function F that takes into account both the length and the number of identities:

$$F = F_{forward} + F_{backward}$$

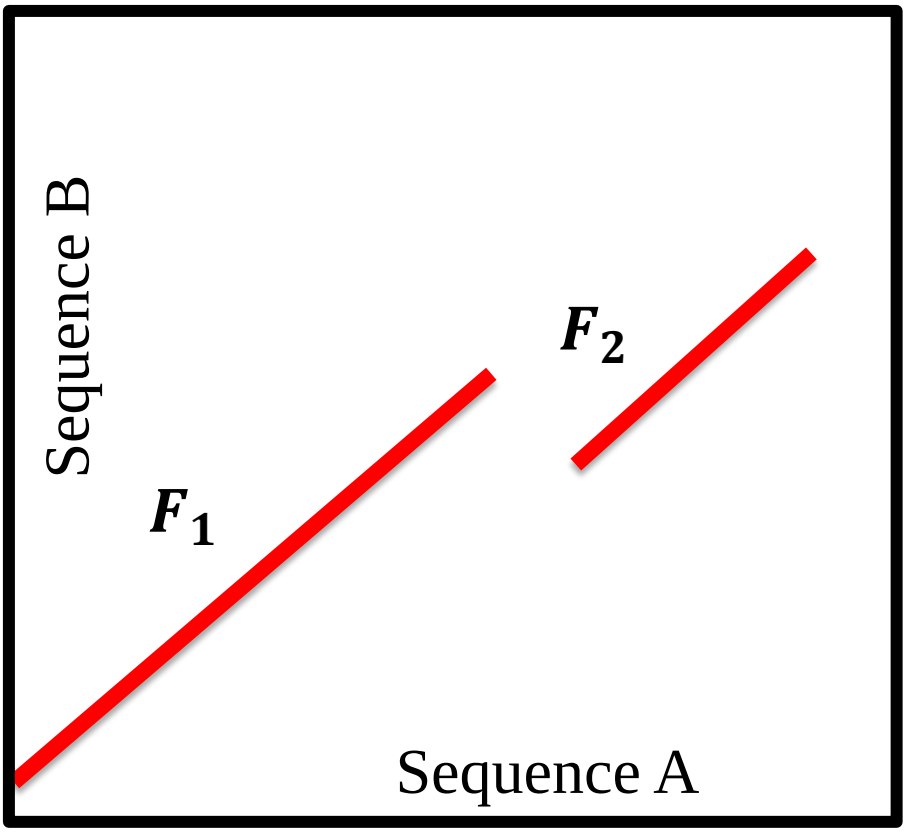
$$F_{forward} = \max \sum_{i=0} PAM(S^A_{k_1+i}, S^B_{k_1+i})$$

Subject to $k_1 + i < n_1,\ k_2 + i < n_2$

$$F_{backwards} = \max \sum_{i=0} PAM(S^A_{k_1-i}, S^B_{k_1-i})$$

Subject to $k_1 - i > 0,\ k_2 - i > 0$

Extending the hits (2)



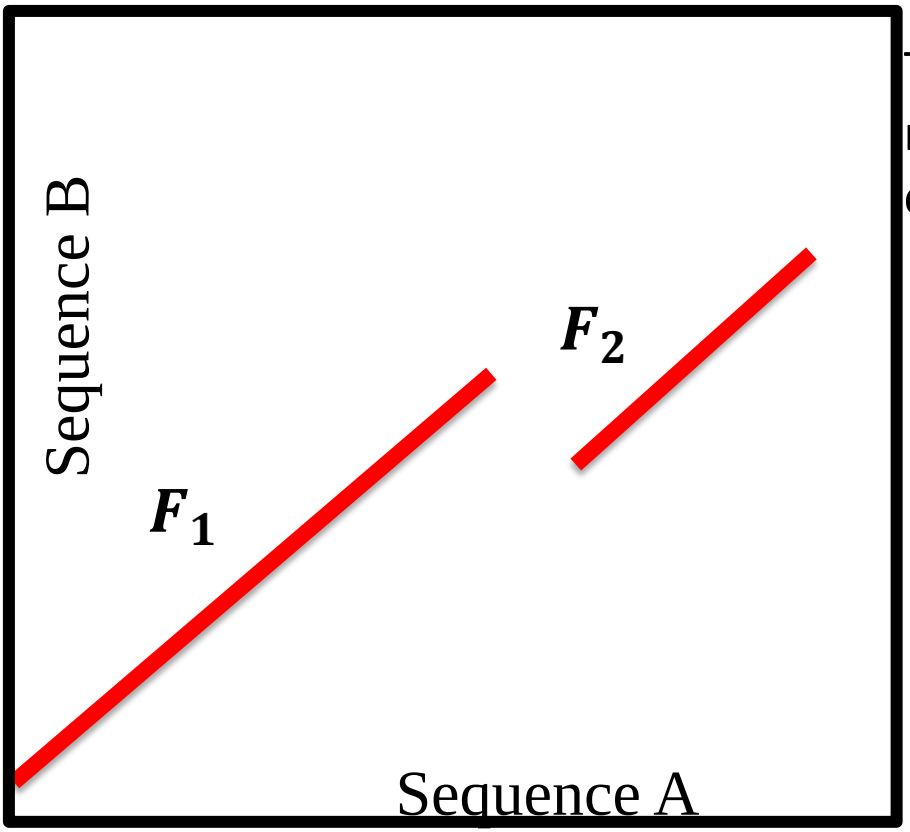
Seed size (k) of 8

```
AAAATACAGTCCCGGCAGGTAGACTTGCGAAACACTTTGATA
|||  |||||||| ||||  | |||     |       |    ||||
AAACTACAGTCCGGGCATGAAGAACAACTTGCGAAACAGATA
```

$$F_1 = H_1 + e_1 \qquad F_2 = H_2 + e_2$$

Hits are now transformed into fragments (which allow mismatches). The fragments will always include the hit plus an extension alignment e_i

Computing the expected value

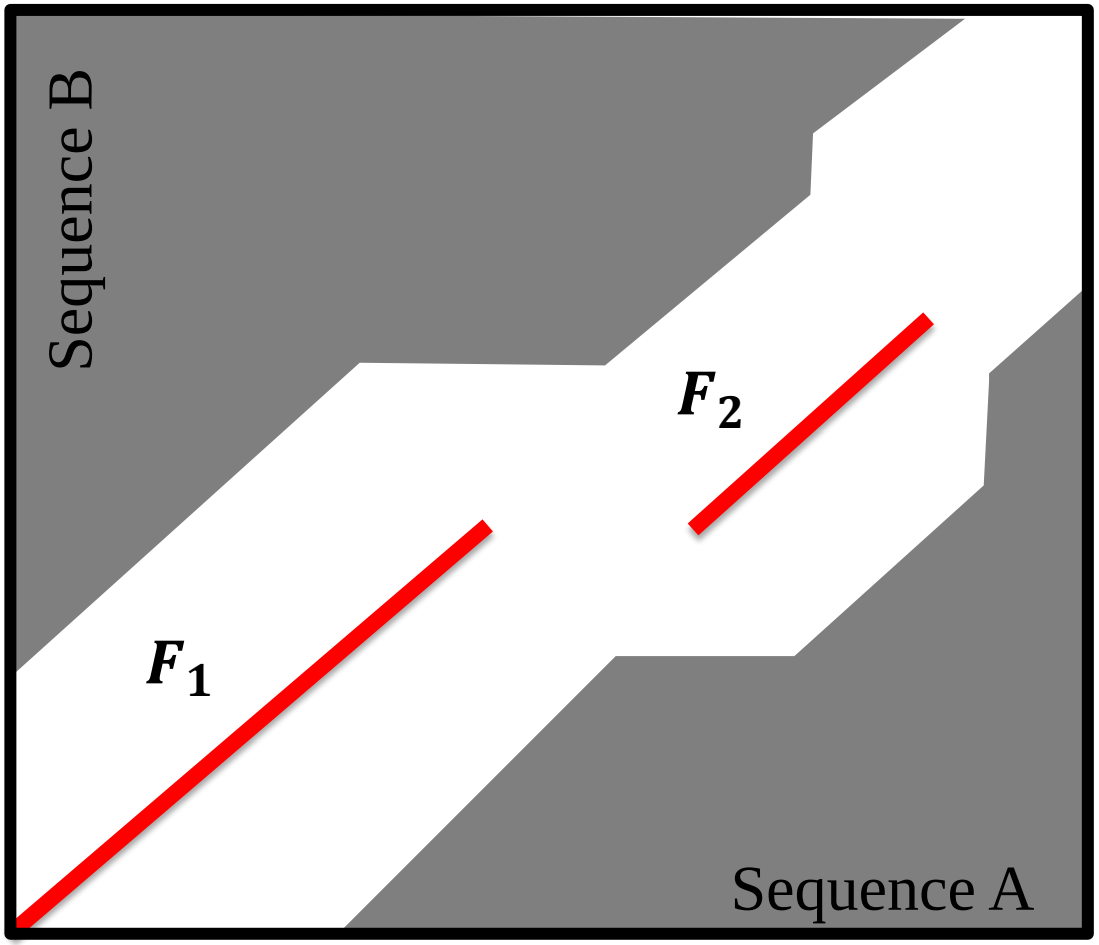


The e-value associated to F_1 is the number of times we would observe F_1 by chance.

$$E = Kmne^{-\lambda S}$$

Where K and λ are the Karlin and Lambda parameters, which are obtained from the scoring matrix (e.g. PAM), m and n are the lengths of query and reference (search space) and S is the raw score for the alignment.

Using fragments as anchors

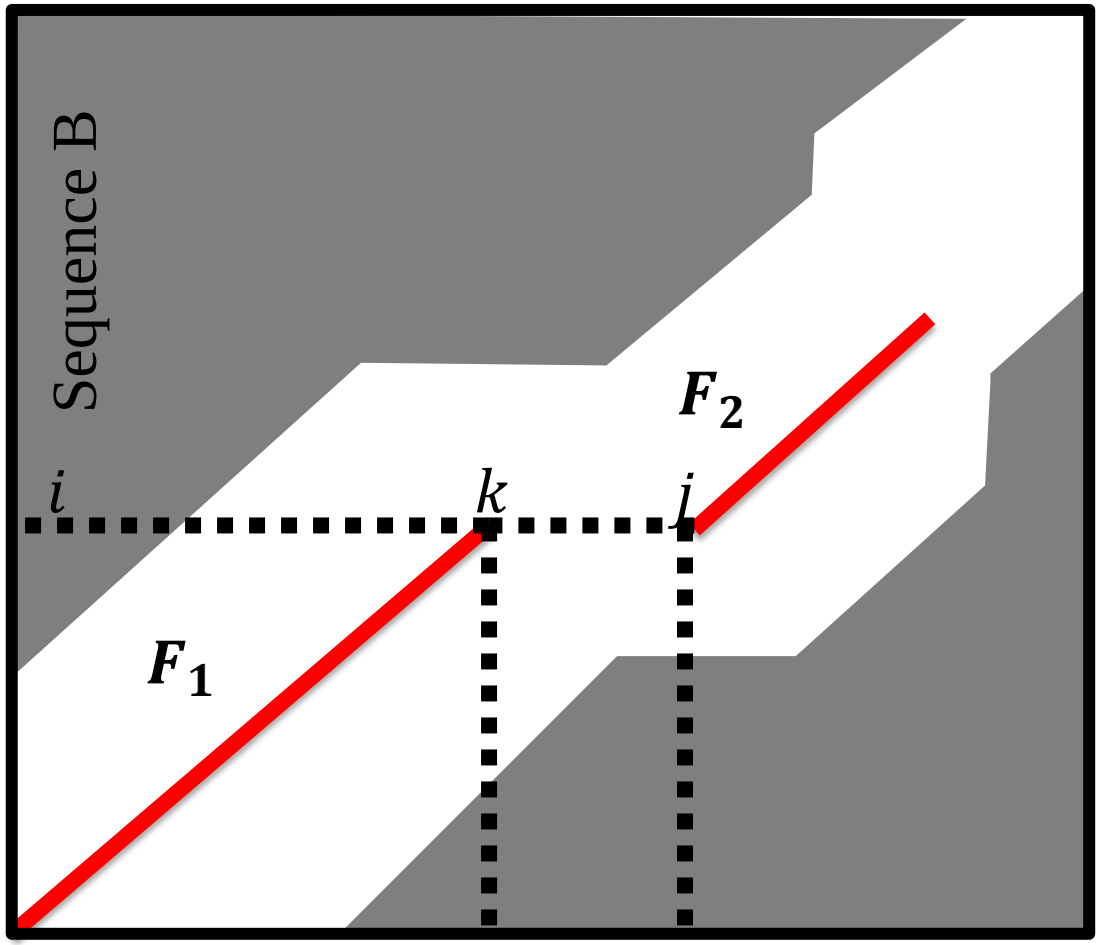


To perform a global Needleman-Wunsch alignment, we can use extended hits as anchors to reduce computation time and space.

Using an affine gap penalty scheme we can model indels, and penalize larger gaps differently from smaller gaps.

Only the anchored diagonal is explored!

Using fragments as anchors (2)



Each cell of the dynamic programming matrix can be calculated as:

Previous diagonal:

$$S(i,j) = PAM(i,j) + S(i-1,j-1)$$

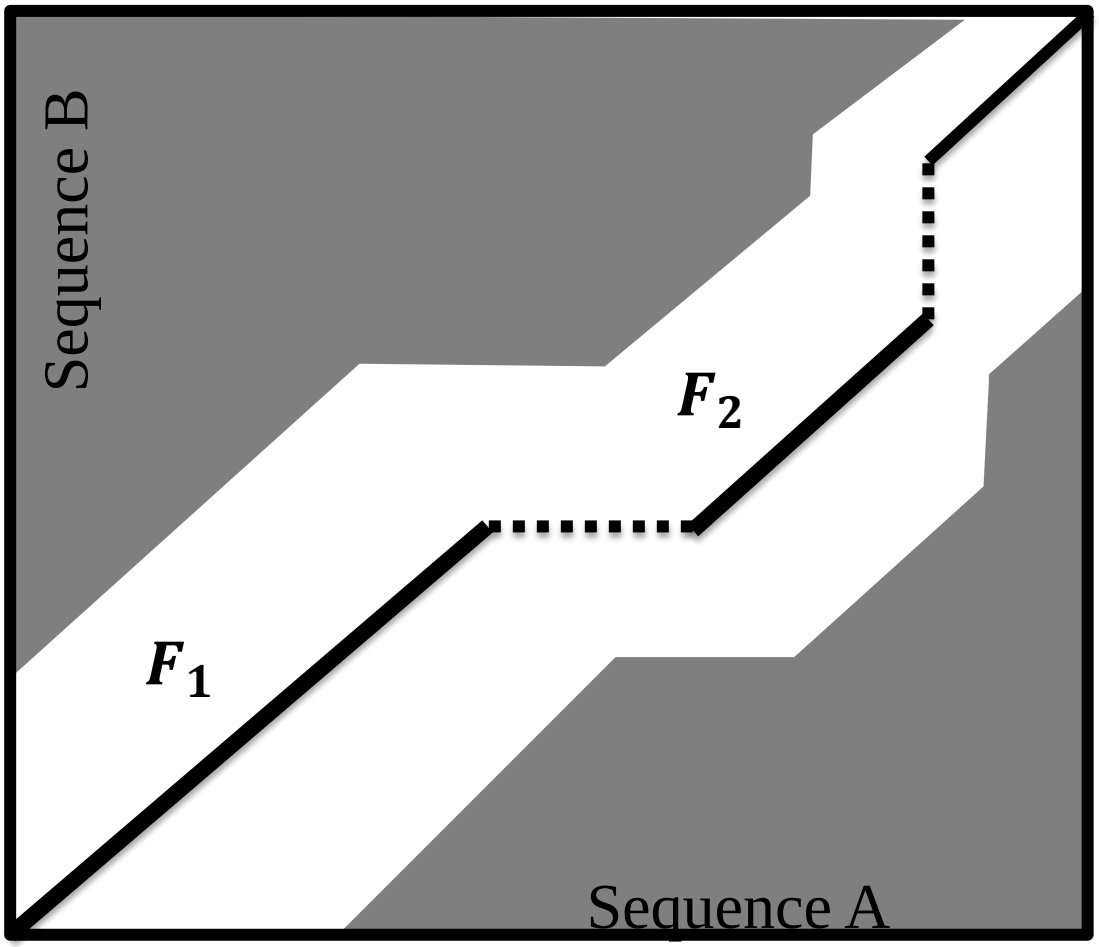
Gap in sequence A:

$$S(i,j) = PAM(i,j) + MAX(r_k + g_{open} + (j-k) * g_{extension})$$

Gap in sequence B:

$$S(i,j) = PAM(i,j) + MAX(c_k + g_{open} + (i-k) * g_{extension})$$

Using fragments as anchors (3)



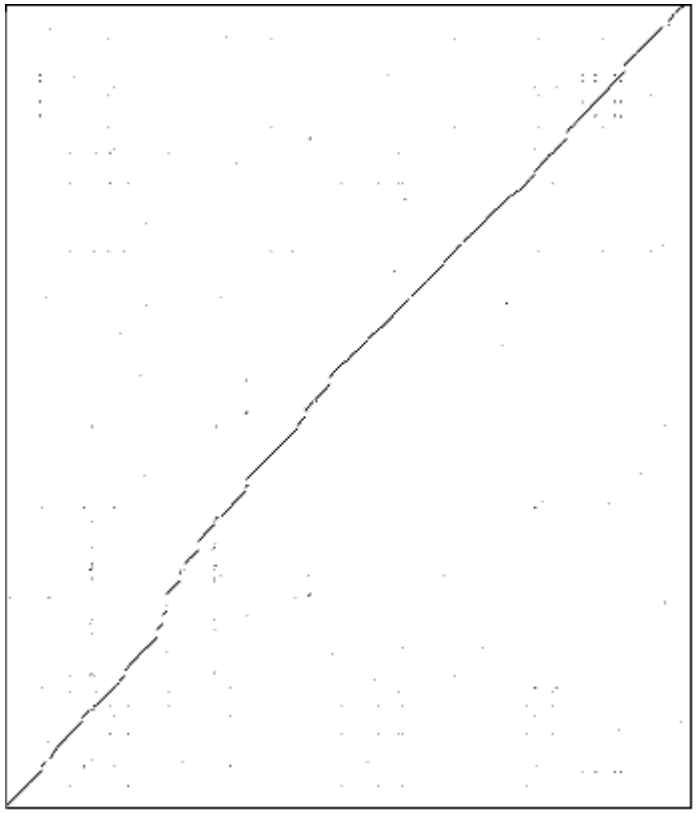
The expected value of a gapped alignment can also be calculated following a similar procedure to that of an ungapped alignment. See [1] for further details.

The resulting alignment would be:

```
AAAATACAGTCCCGGCAGGTAGA----CTTGCGAAACACTTTGATA
|||  ||||||||  ||||  |  |||    ||||||||||    ||||
AAACTACAGTCCGGGCATGAAGAACAACTTGCGAAACA----GATA
```

1. Altschul, Stephen F., et al. "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs." *Nucleic acids research* 25.17 (1997): 3389-3402.

E.coli (K12 vs O157) dotplot (≈ 5 Mbp)



		Time (sec)		
Process	**Input**	**User (u)**	**System (s)**	**Elapsed**
maskSeq	ECk12.fa	0,164	0,024	0,188
maskSeq	hits32	0,196	0,036	0,232
words	ECk12m.fa	0,736	0,196	0,932
words	ECO157m.fa	0,884	0,216	1,100
sort	ECk12.words.unsort	1,600	0,412	2,012
sort	ECO157.words.unsort	1,108	0,372	1,480
w2hd	ECk12.words.sort	1,124	2,428	3,552
w2hd	ECO157.words.sort	1,280	2,916	4,196
hits	ECk12 ECO157	1,360	0,740	2,100
sortHits	EChits32	3,012	0,968	3,980
FragHits	EChits32.order	0,380	0,032	0,412
				20,184

Sequence length	DOTTER	Gepard	Gepard pre-SA
10 000 bp	2 s	<1 s	<1 s
50 000 bp	30 s	<1 s	<1 s
100 000 bp	2 min 4 s	<1 s	<1 s
1 000 000 bp	2 h 10 min	5 s	4 s
5 000 000 bp	52 h 38 min[a]	47 s	40 s
Human chrom. I	382 years[a]	61 min	53 min

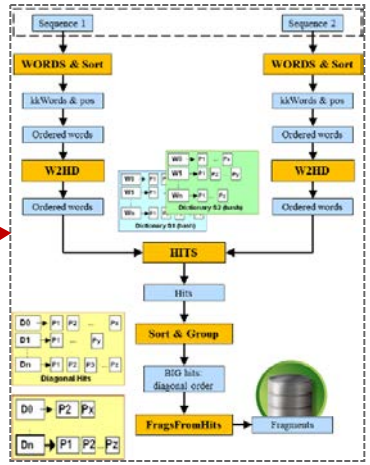
Reference: Krumsiek, Jan, et al. (2007); "**Gepard**: a rapid and sensitive tool for creating dotplots on genome scale"; Bioinformatics Vol. 23 no. 8,

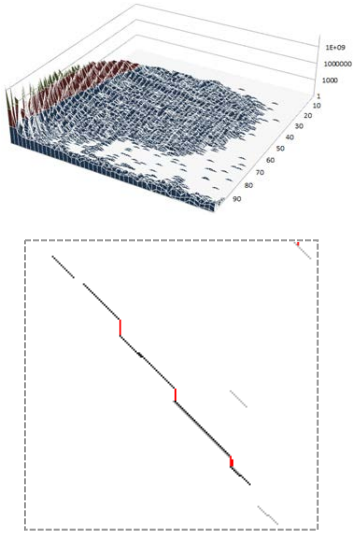
… in 20 seconds

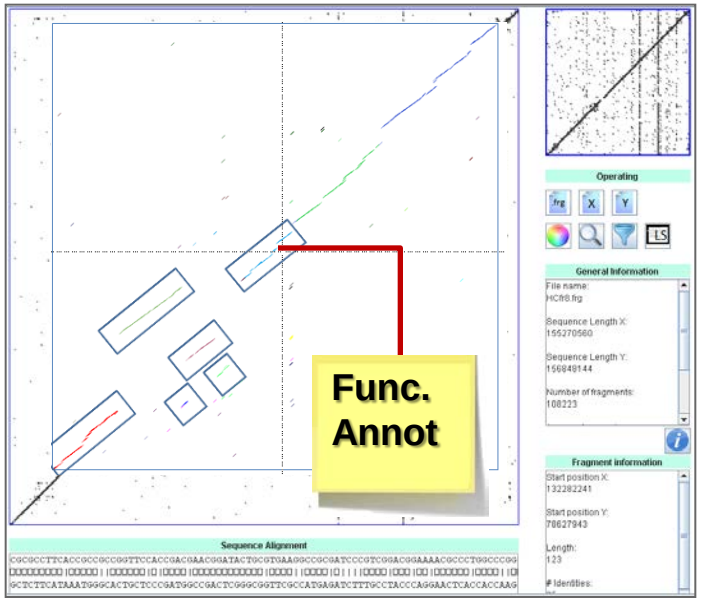
Mammalians vs Human Chr$\mathcal{X}$ (≈ 150 Mbp)

10 mins x pairwise gc (30h per full mammalian genomes) → HPC & Cloud

Comparative genomics: Challenges





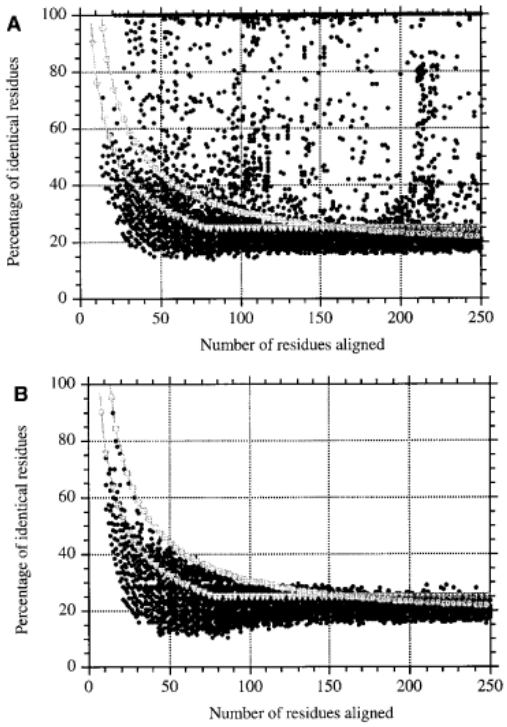


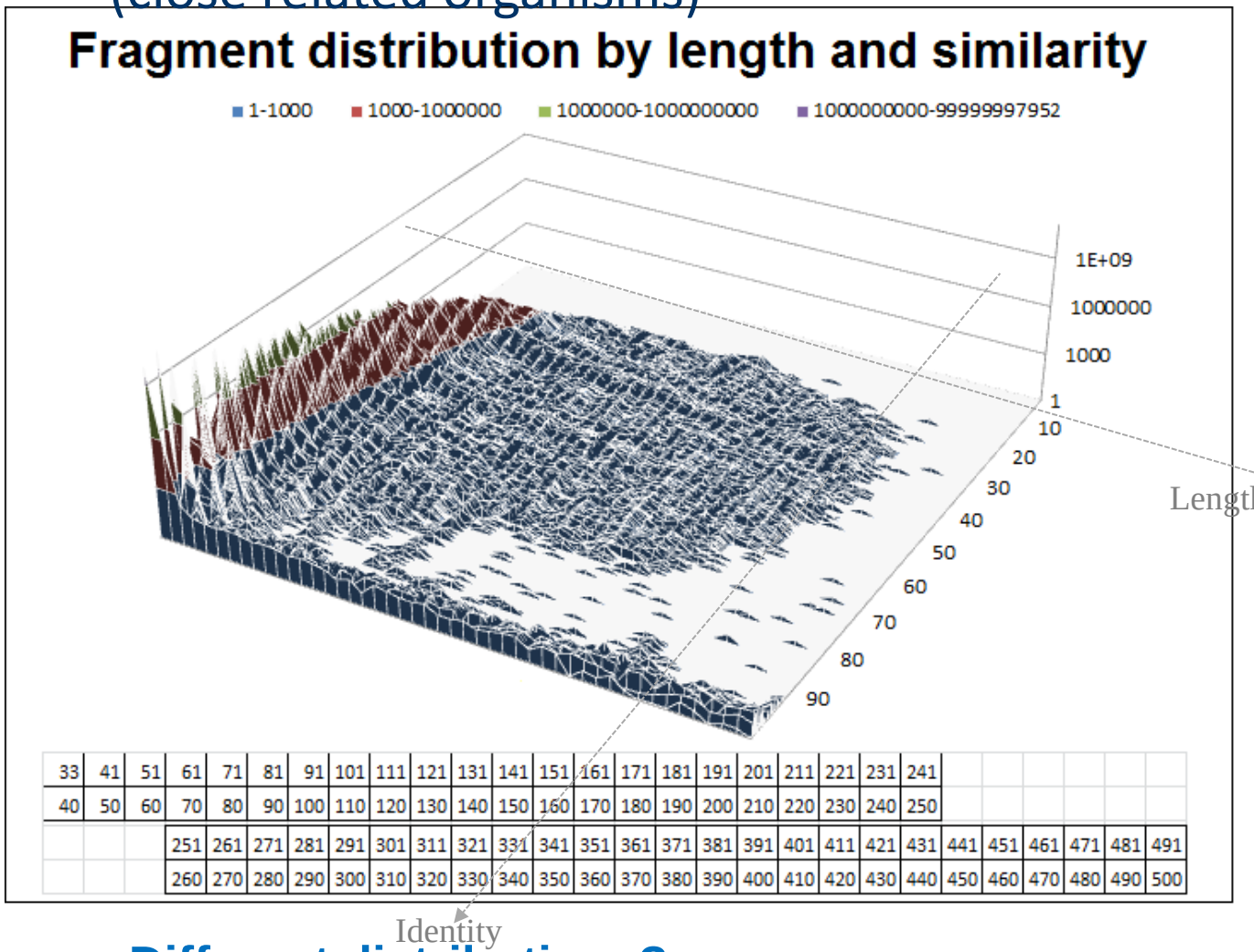
Big-data; HPC + cloud; new comparative models, interactive Visualization for Data analysis; GUIs; mobile devices

The End

Human vs Chimpanzee

(close related organisms)





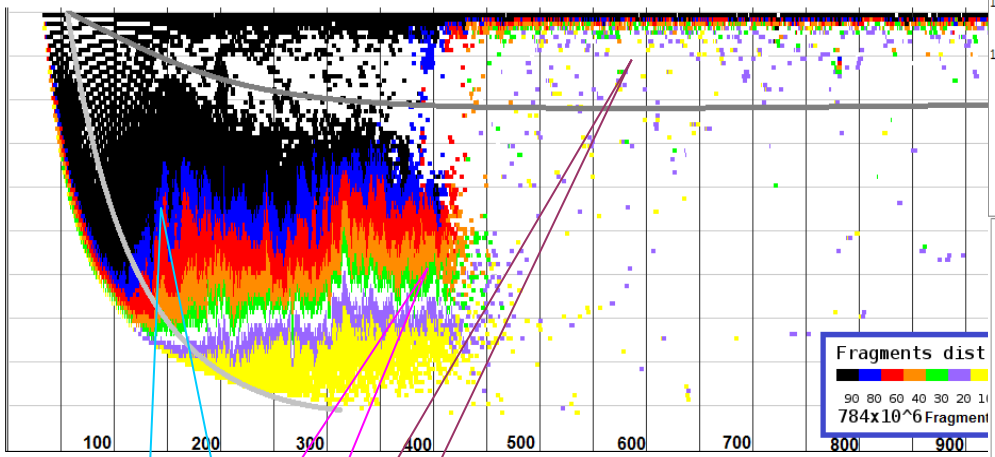
Different distributions?

Introns, exons, intergenic… produce the same type of fragments?

Statistical Significance of HSPs

New models are needed

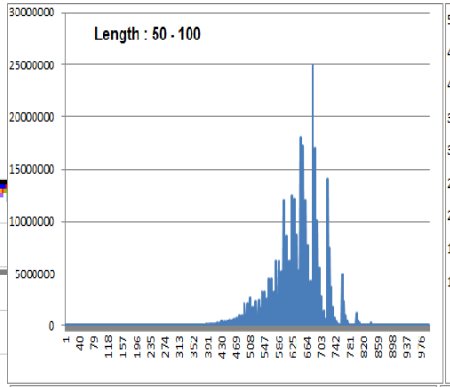
Coverage Levels

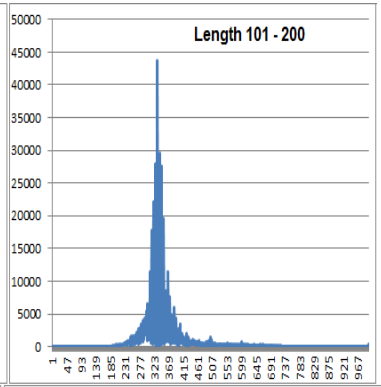


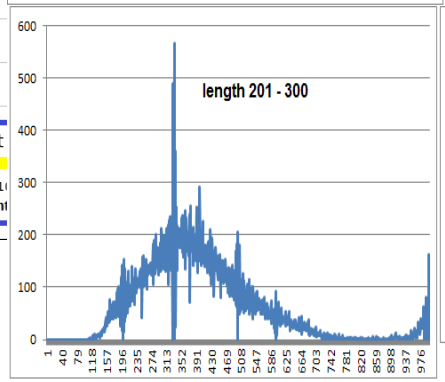
random

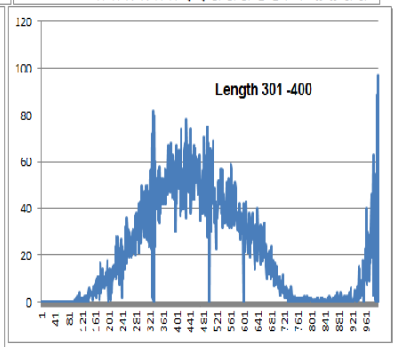
intron / intron

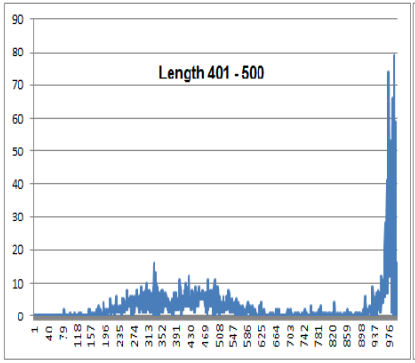
Exon / exon

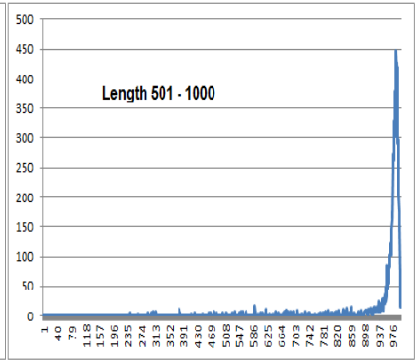




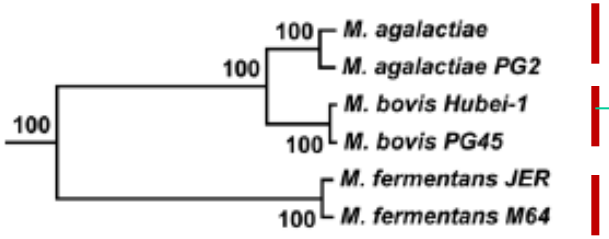


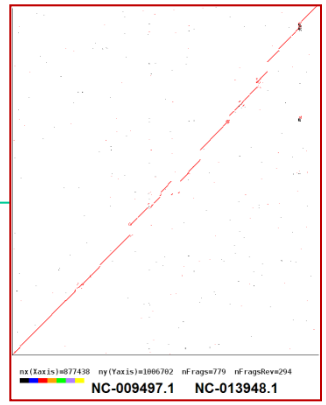


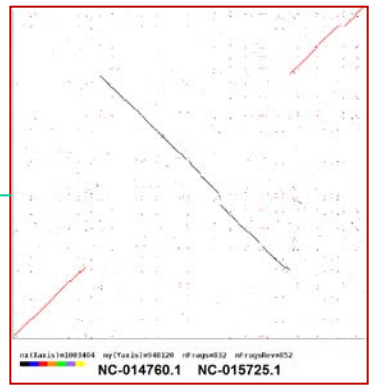


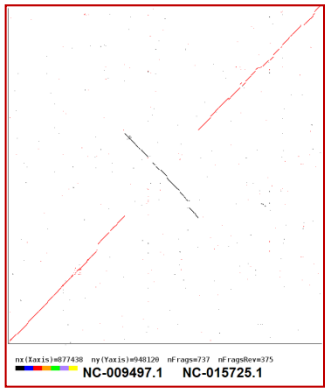


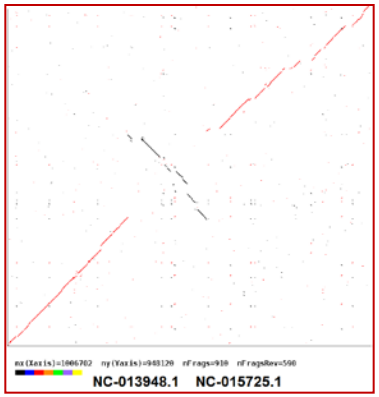
Multiple Genome comparison & phylogeny

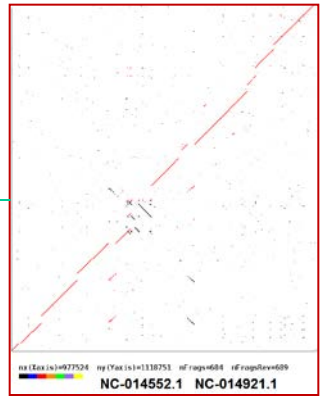








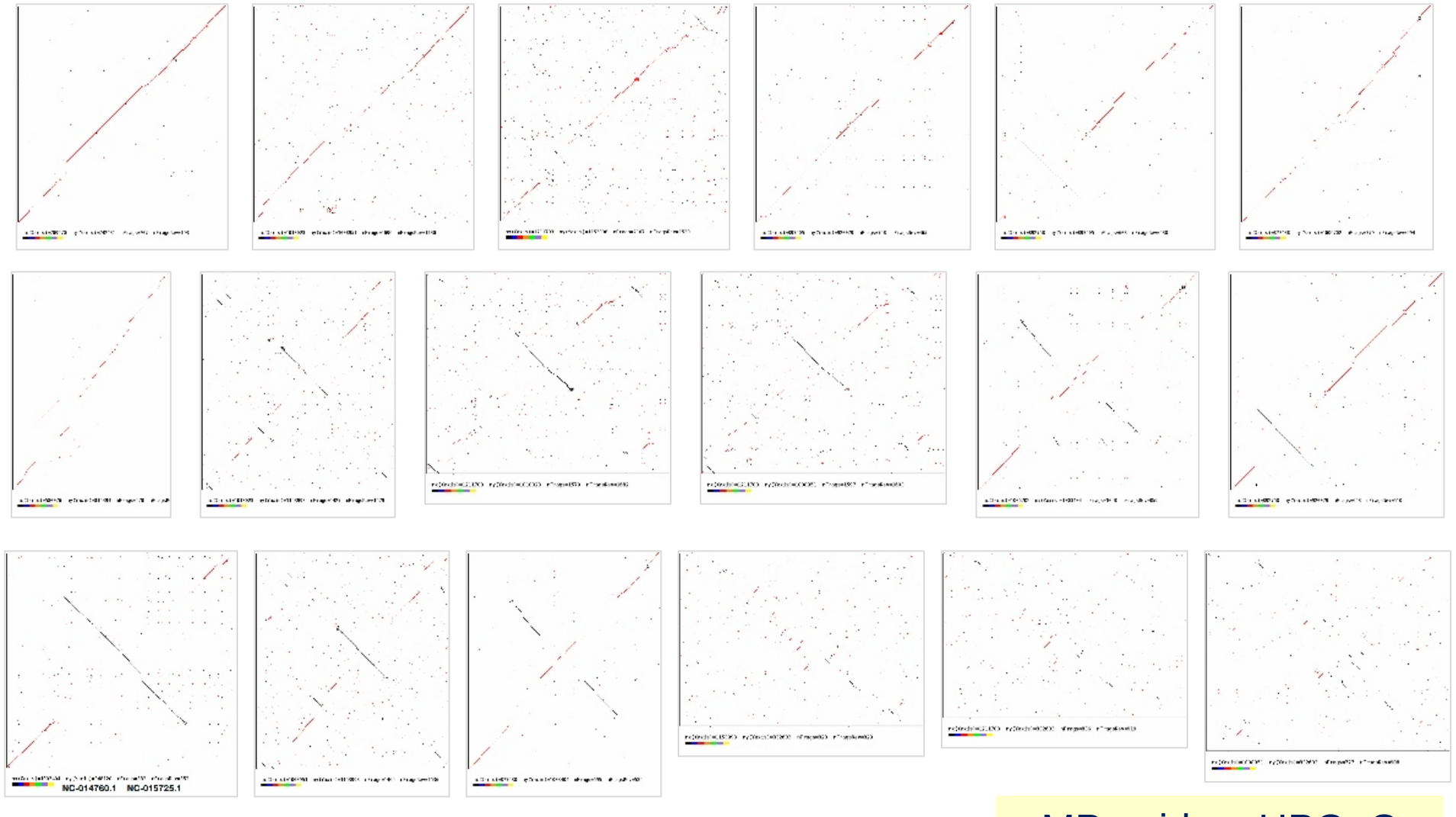




Pairwise genome alignment
DB searchin strategy
→ Big-Data Kmers dictionaries
Workflows
Visualization

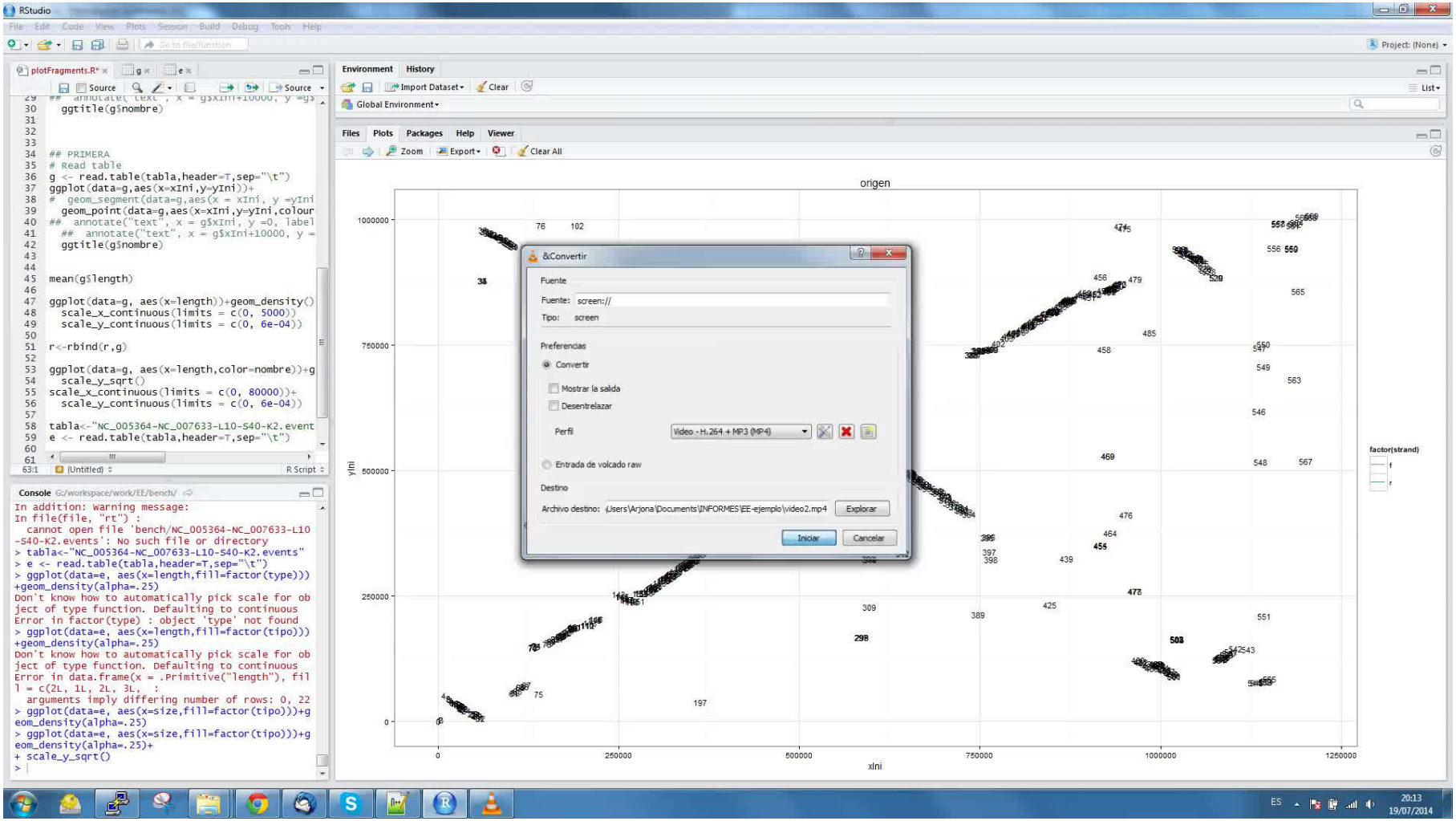
Modeling
Blocks detection + refinement
Breakpoints identification
EE frequencies
Inter-genome distances
GUis

O.Trelles, PhD



MProvide a HPC+C suite for highly demanding CG studies

The intermediate species



Memory demand

Genome Size	d2hP		d2hW		hits		hits reverse		frags		frags.INF	
	value	ratio	value	ratio	value	ratio	value	ratio	vaue	ratio	value	ratio
1017820	16053968	15,773	22430016	22,037	75520	0,074	51560	0,051	32287	0,032	524	0,001
2307087	36391216	15,774	53444088	23,165	5040	0,002	4760	0,002	1132	0,000	534	0,000
5577086	87974704	15,774	127753872	22,907	3065440	0,550	205560	0,037	135796	0,024	536	0,000
32580616	521203904	15,997	779044512	23,911	14550600	0,447	9521960	0,292	3117655	0,096	527	0,000
252513033	3687610544	14,604	5016212256	19,865	35925818920	142,273	33571929880	132,951	3706903	0,015	549	0,000
252513033	3687610544	14,604	5016212256	19,865	25534140600	101,120	25506710400	101,011	1362745	0,005	549	0,000
252513033	3687610544	14,604	5016212256	19,865	692532400	2,743	690769960	2,736	109	0,000	541	0,000
304123439	3687610544	12,125	5016212256	16,494	19209800	0,063	12965680	0,043	60838	0,000	538	0,000
160599122	2514782336	15,659	3309071016	20,605	19094190000	118,893	18939227360	117,929	200059	0,001	547	0,000

frags.MAT		csv		comparison size est. (MB)	comparison size real (MB)	Difference
value	ratio	value	ratio			
201021	0,198	24224	0,024	113,38	110,00	3,08%
201005	0,087	1347	0,001	263,83	241,71	9,15%
201097	0,036	102168	0,018	636,70	600,00	6,12%
204548	0,006	2601235	0,080	3841,86	3891,20	-1,27%
363403	0,001	3122387	0,012	91909,36	91136,00	0,85%
345777	0,001	1075438	0,004	74303,29	73324,04	1,34%
215406	0,001	568	0,000	26943,71	23865,22	12,90%
202494	0,001	49858	0,000	25802,93	17749,72	45,37%
375495	0,002	160378	0,001	53393,85	52642,85	1,43%