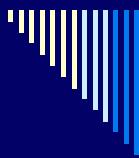
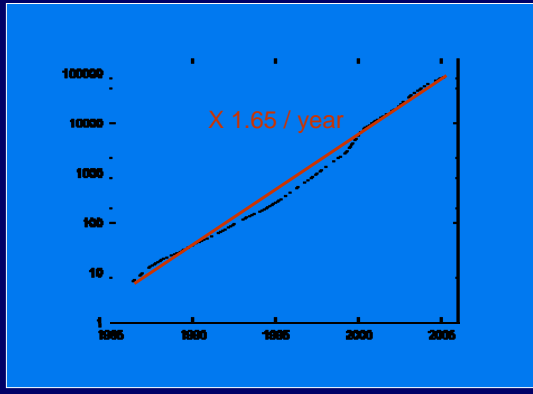


Fine-Grained Parallel Genomic Computation

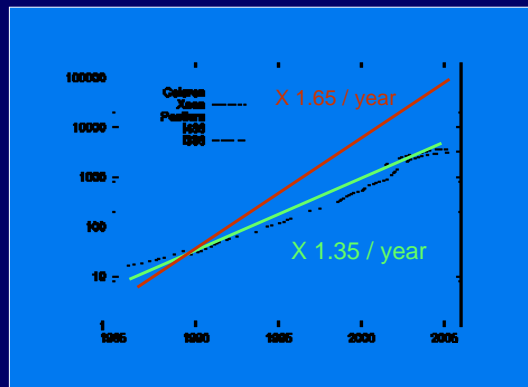
Dominique Lavenier
CNRS / IRISA
Rennes, France



GenBank *nucleotide database*



Microprocessor Performances



02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
 France

3

Data vs proc. performances

2006 : GenBank 120 Gbp
 computational power: 12 Gbp/sec
 → time: 10s

2010 : GenBank 900 Gbp
 computational power: 40 Gbp/sec
 → time: 25s

2014 : Genbank 6.6 Tbp
 computational power: 130 Gbp/sec
 → time : 50s

1.65^4
 7.4x
 1.35^4
 3.3x

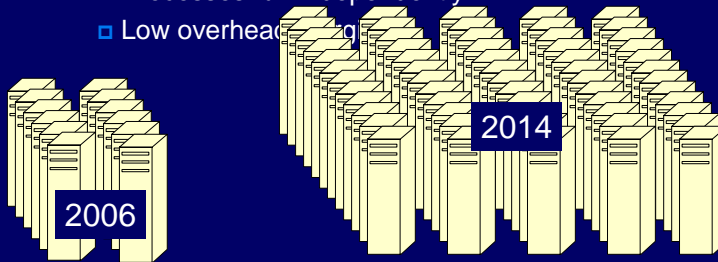
02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
 France

4

Genomic processing

- High potential parallelism
 - Very efficient on large clusters
 - Data are dispatched on the nodes
 - Processes run independently
 - Low overhead

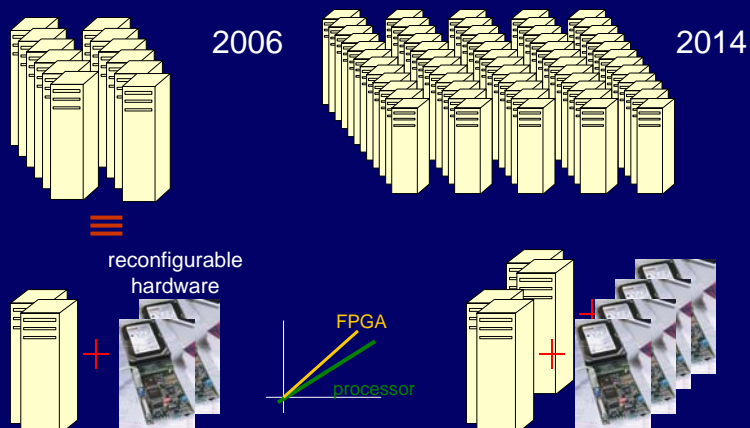


02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

5

Fine-grained processing



02 23 2006

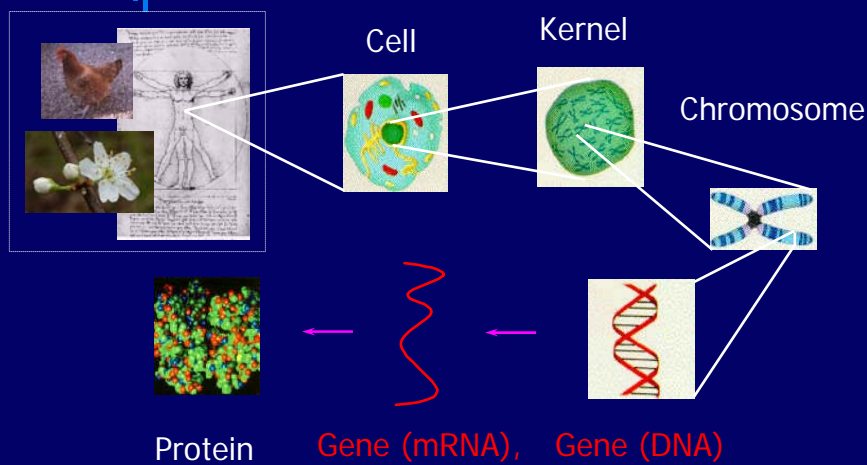
D. Lavenier - CNRS/ IRISA - Rennes
France

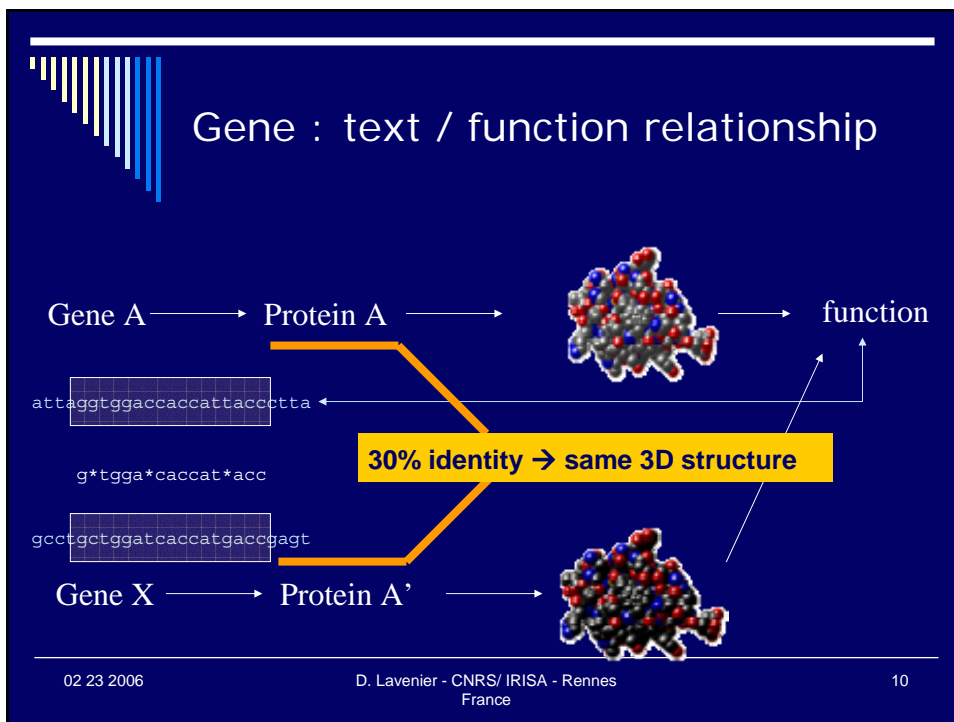
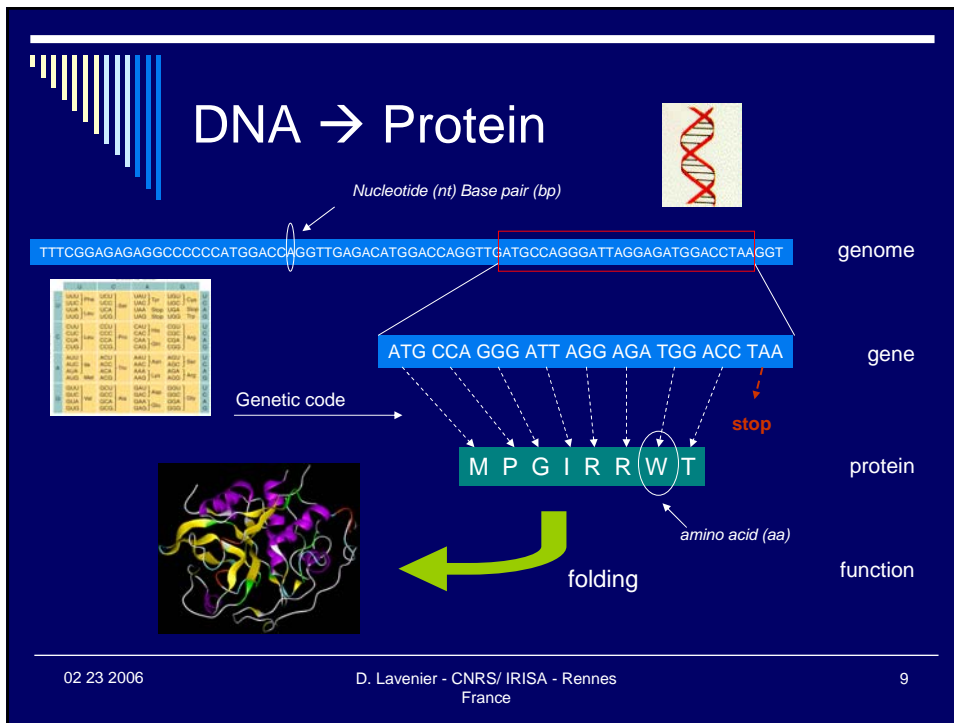
6

Content

- Molecular biology
- Genomic processing examples
- Algorithms
 - Dynamic programming
 - Seed-based
- Hardware

Organism → molecule

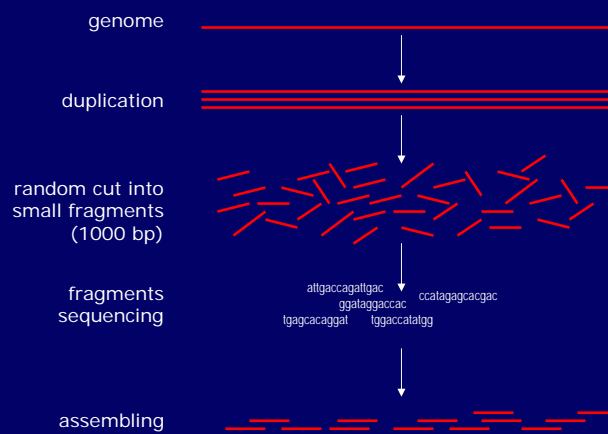


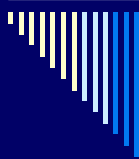


Content

- Molecular biology
- Genomic processing examples
 - Shotgun sequencing
 - Content-based search
 - Motif detection
- Algorithms
- Hardware

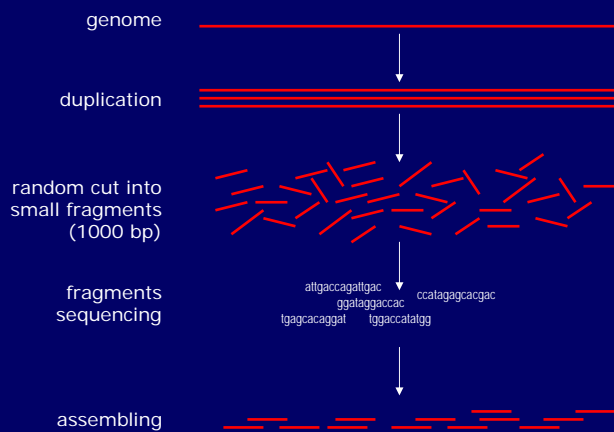
Example 1 *Shotgun sequencing*





Example 1 Shotgun sequencing

Human



3×10^9
genome size

10 X
coverage

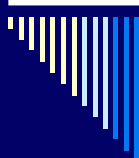
3×10^7
number of fragments

5×10^{15}
pairwise comparisons

02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

13



Pairwise comparison

substitution error insertion/deletion error



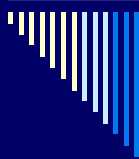
errors:

- sequencing machines are not perfect
- polymorphism (when a few individuals are sequenced together)

02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

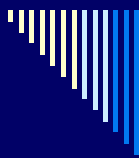
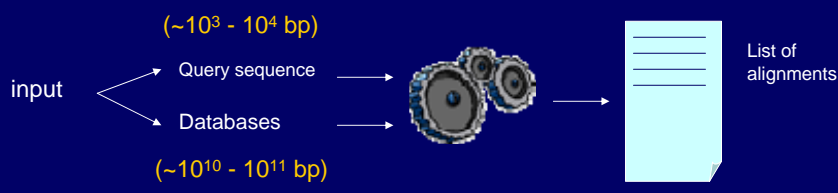
14



Example 2 Content-based search

unknown gene ?

find in the genomic databases genes having similar text



Alignments

Sequence #1

ATTAGACACAGGATATAGAGCACACAGGATTAGGACCAGGGATTAGGGAGAGGACTAAATATAGAGCATATAGGAGGTAGGATCCAGGGATTAG

Sequence #2

TGGACCAGATTAGAGCCAGGATATTAGAGGATATAGGAAGGTAGGACCGGTAGACCAGTTAGACCAGGGATTTAGGGGGATTAGGAGTTAAA

Finding a local alignment
 we don't know:
 - its length
 - its position in both sequence

TAGAGCATATAGGA . GGTAGGATCC
 ||||| ||||| ||||| ||||| |||||
 TAGAGCATATAGGAAGGTAGGA . CC

Alignment scoring

$$\text{Score} = 21 \times 1 - 2 \times 3 - 2 \times 2 = 11$$

```

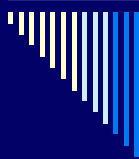
TAGAGCATATAGGA.GGTAG AATCC
| | | | | | | | | | | | | | |
TAGAGGATATAGGAAGGTAGGA.CC
  
```



Necessary to compare alignments → extract the best ones

Content-based search

- Activity #1
 - May represent more than 70 % of bioinformatics computation
- NCBI (*National Center for Bioinformatics*)
 - Handle more than 100,000 queries each day
- Have to deal with
 - The exponential growth of the databases
 - The increasing user needs



Example 3 Motif detection

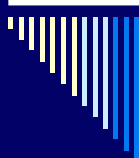
- Gene families may contain specific signatures
- A signature is a pattern of amino acids (aa)
- Example :

■ M $x(1-3)$ K P L [I V L] $x(2-4)$ [R S] L

MTEKPLVSTRL is valid



- Can be expressed by regular expressions



Complex motifs

- 2 D structure

A A
T C
A-T
C-G
G-C
A-T

ATGCTGCTGAT GCTGAAC

- Can be expressed by string variable grammars

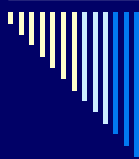
- Example

■ ATGCTGCTGATAGCATAACTGCTGCTGAAC

repeat

bio-palindrome

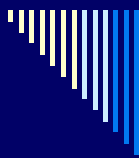
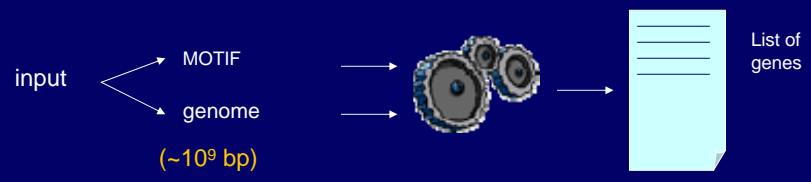
Y:3 Y x(2-6) X:4 x(3-6) \bar{X} x(4) AA



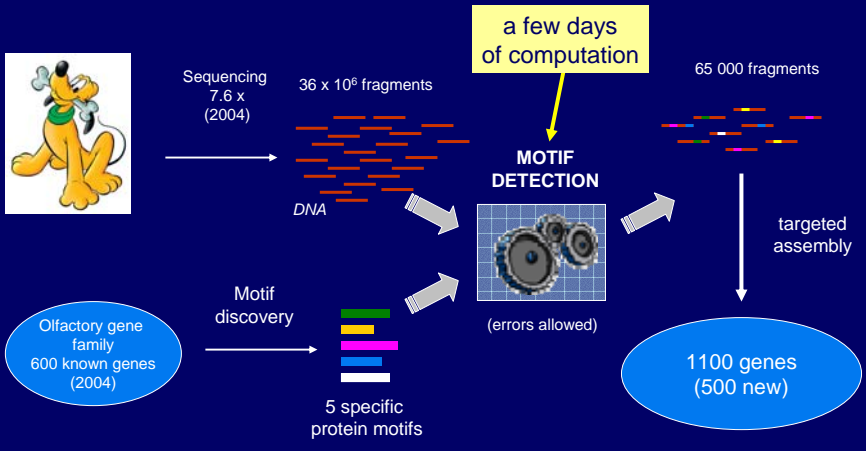
Motif search

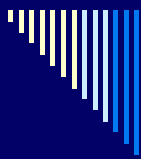
signature

find in the genomic databases genes having this signature



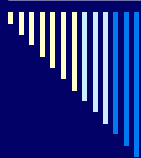
Dog olfactory gene discovery





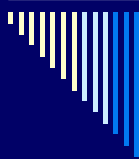
Genomic Processing features

- Data
 - Sequences
 - DNA (4 letters) – protein (20 letters)
 - Size : 10-100 Gbp (*1 Tbp* → 2010)
- Algorithms
 - Mostly string processing
 - approximate sequence comparison
 - Integer operation
 - Reduced arithmetic (8 – 16 bits)



Content

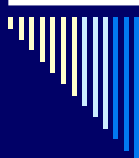
- Molecular Biology
- Genomic processing examples
- Algorithms
 - Dynamic Programming
 - Seed-based
- Hardware



Dynamic programming

- Input : 2 (text) sequences
- Output : 1 alignment
- Idea:
 - Find the minimal number of operations to move from sequence #1 to sequence #2
- Needleman & Wunsch (1970)
 - Global alignment
- Smith & Waterman (1981)
 - Local alignment

GLOBAL *substitution*
GLOCAL *omission*
.LOCAL
LOCAL



Dynamic programming

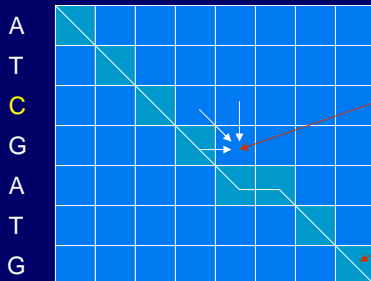
Needleman & Wunsch – Global alignment

Sequence #1 (size N)

A T A G A C T G

Complexity : N^2
($N = \text{sequence length}$)

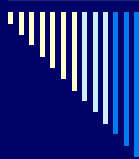
Sequence #2 (size N)



$F_{\text{Global}}(M_{i,j-1}, M_{i-1,j-1}, M_{i-1,j})$

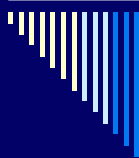
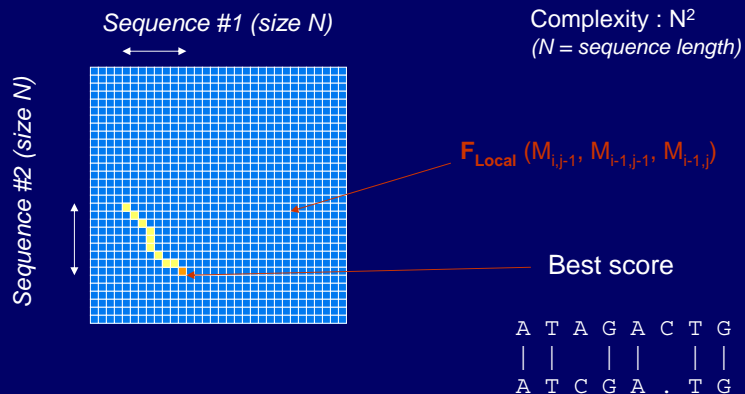
Score of the alignment

A T A G A C T G
| | | | | |
A T C G A . T G

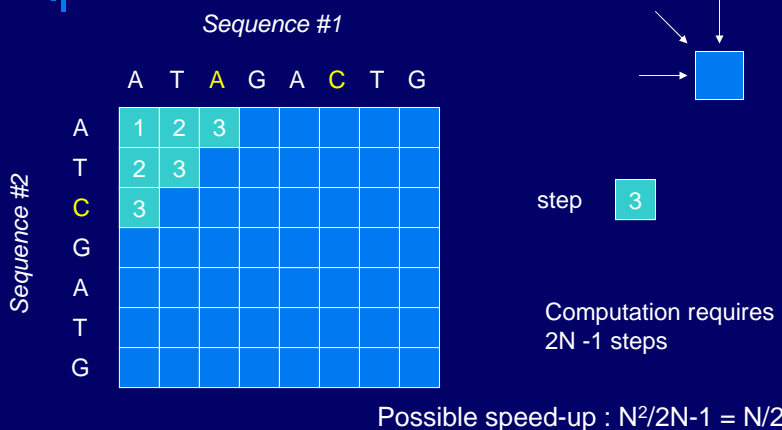


Dynamic programming

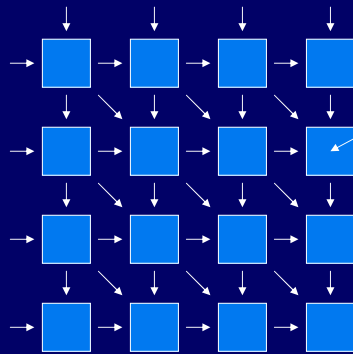
Smith & Waterman – Local alignment



Data dependencies



2D systolic architecture



1 systolic cell performs

$$F(M_{i,j-1}, M_{i-1,j-1}, M_{i-1,j})$$

$$N^2 \text{ cells} \rightarrow \text{speed-up} \frac{N^2}{2N-1} \sim N/2$$

Bad efficiency

02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

29

1D systolic architecture

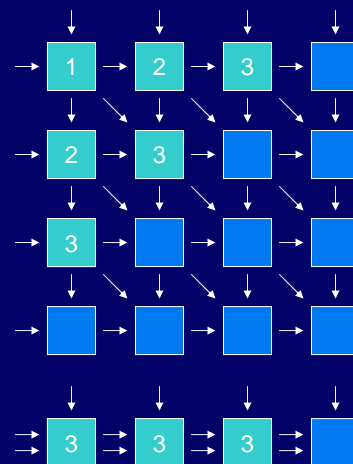
2D array :
- only one diagonal active at each step

1D array
- one cell emulates one column

step 3

N cells \rightarrow speed-up $\sim N/2$

better efficiency than 2D structure

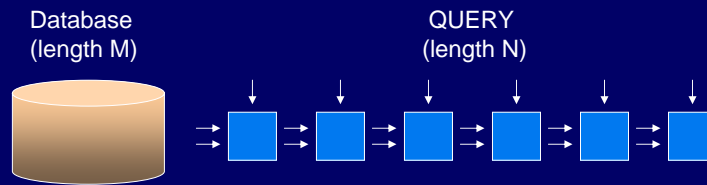


02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

30

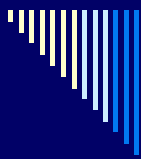
Database scanning



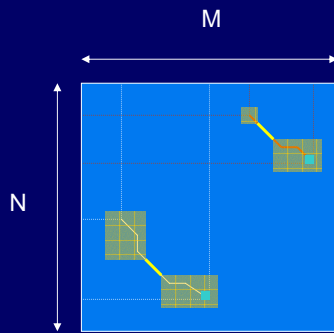
$$\text{Speed-up} = \frac{N \times M}{N + M - 1} = \sim N \quad (N \ll M)$$

Dynamic Programming *Conclusion*

- Advantages
 - Gives the exact solution
 - Efficient parallelization on 1D systolic architecture
- Drawback
 - Too high complexity (N^2) to handle large databases with sequential machines
 - Scan of GenBank can take a few hours
- Need of heuristic to decrease computation time
 - FASTA (*Pearson, Lipman 1988*)
 - BLAST (*Altschul et al. 1990*)



Seed-based algorithm *idea*



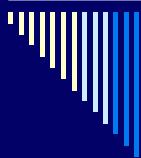
Local Alignments

Programming dynamic:

1. Full matrix computation
2. Highest scores detection
3. Trace-back

Seed-based idea:

1. find "high similarity" regions from seeds
2. limit the search space near these regions

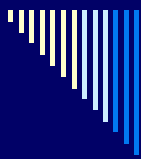


Seeds ?

- A common word of K characters includes in both sequence

ATTAGGACCA**TTAGGC**AGGACCTCCAGGATAGGACCCAGGAGTTAGAC

TTTAGGACACGAGGATATGGACCAGGCC**TTAGGCT**GGACCGTGTTAG



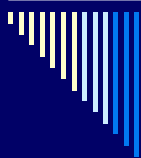
Seed ?

- A common word of K characters includes in both sequence

ATTAGGACCATTAGGCAGGACCTCCAGGATAGGACCCAGGAGTTAGAC

TTTAGGACACGAGGATATGGACCAGGCCCTTAGGCTGGACCGTGTTAG

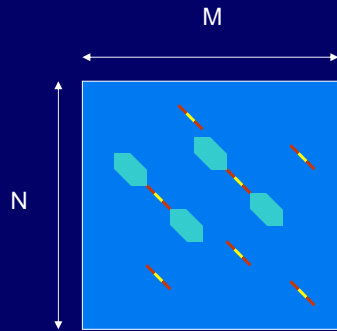
```
GACCATAGGCAGGACC . T
| | | | | | | | | | |
G . CCCTAGGCTGGACCGT
```



Seed based algorithm efficiency

- 100X – 1000X time faster than Programming Dynamic
 - Space search is drastically reduced
→ default k-word size = 11
- Large databases can be scanned in a few seconds
- Is there still room for fine-grained parallelism ?

Seed based algorithms



- k-word hit
- Ungapped alignment
if score > threshold
- gapped alignment

Profiling

~ 85 %

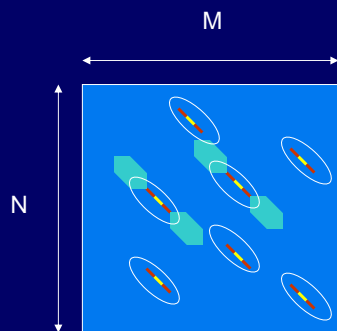
~ 12 %

02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

37

Parallelization of seed based algorithms



2 sub levels of parallelism

1 multiple gapped/ungapped alignments can be performed simultaneously

→ high data throughput

2 alignment computation

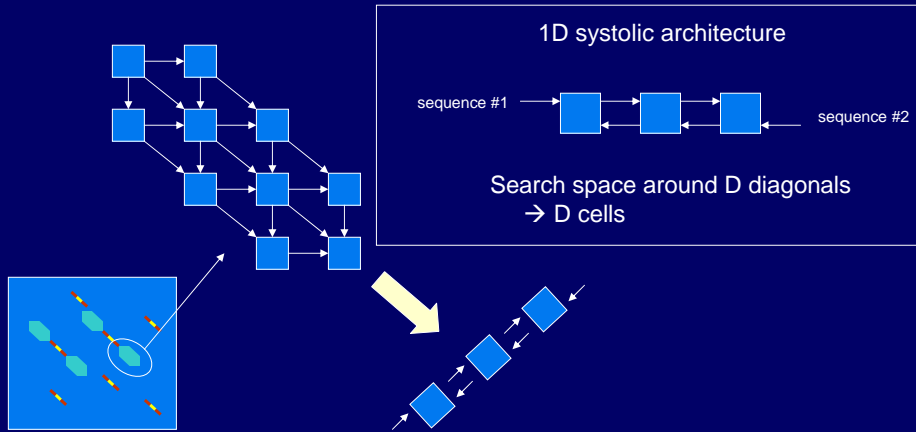
→ DP on a restricted area

02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

38

Dynamic programming on a restricted area



02 23 2006

D. Lavenier - CNRS/ IRISA - Rennes
France

39

Seed-based algorithms

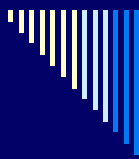
Conclusion

- Very powerful heuristic
 - *BLAST software can compute hundred of Mbp/sec*
- Adopted by the scientific community
 - *BLAST is the reference software*
- High potential for fine-grained parallelism
 - However: more complicated than dynamic programming (data access)

02 23 2006

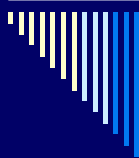
D. Lavenier - CNRS/ IRISA - Rennes
France

40



Content

- Molecular Biology
- Genomic processing examples
- Algorithms
 - Dynamic Programming
 - Seed-based
- Hardware



2 main classes of applications

- Intensive computation
 - Shotgun
 - Genome structure analysis
 - Ex: Repeat sequences
 - Genome comparison
 - Ex: Human \leftrightarrow mouse
- Database querying
 - One (small) query against the databases

must be done in a few secondes

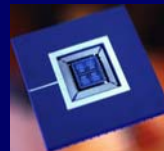
must be done as fast as possible

Clusters

- Coarse grained parallelism
- Well suited for intensive computation
 - Large tasks can run independently
 - Limited I/O operations : data are intensively re-used
- Database querying:
 - Data access can be a bottleneck

VLSI Accelerators

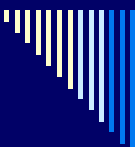
- A few realizations essentially for speeding up DP algorithm (systolic array)
- Advantage
 - Very fast: 50X – 100X
- Drawback
 - Chip need to be periodically redesigned
 - Expensive, niche market
 - Lake of flexibility
 - S&W algorithm





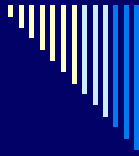
FPGA Accelerators

- Suppress the VLSI drawback
 - Standard components
 - Can be reconfigured indefinitely
- Commercial products available
 - Ex: Decypher Engine from TimeLogic Inc.
 - S&W algorithm
 - BLAST-like algorithm
 - HMM motif search
 - . . .
- Performances: 50X – 100X



Hardware accelerators General *conclusion*

- Fine-grained parallelization
- Speed-up: 50 to 100 compared to microprocessors
- Can be mixed with clusters
 - *i.e. node = 1 processor + 1 accelerator*
- Well suited for intensive computation
- Database querying
 - Do not solve the data access problem



Querying databases

- First bioinformatics activity
- Genomic databases grow faster than
 - processor performances
 - data access time (disks!)
- Solutions
 - Keep the data in main memory and parallelized the accesses
 - *Databases are dispatched on memory cluster nodes*
 - Design a large parallel fast access memory