Introducción a la Bioinformática:

Comparative Genomics: Sequence Alignments

Luis Garreta

Doctorado en Ingeniería
Pontificia Universidad Javeriana – Cali

March 8, 2017

Motivation:

Evolutionary History of the Sequences

Any alignment between two or more nucleotide or amino acid sequences **represents an explicit hypothesis** regarding the evolutionary history of those sequences.

Motivation:

Comparisons of Sequences facilitate their Understanding

Comparisons of related protein and nucleotide sequences have facilitated advances in understanding the content and function of genetic sequences.

Motivation:

Solving Key Problems in Bioinformatics

Sequence alignments provide important information for solving many of the key problems in bioinformatics including:

- Find evolutionary relationships between organisms (genes, proteins), and
- Identify the function of a newly discovered genetic sequence;
- Predicting the structure and function of proteins.

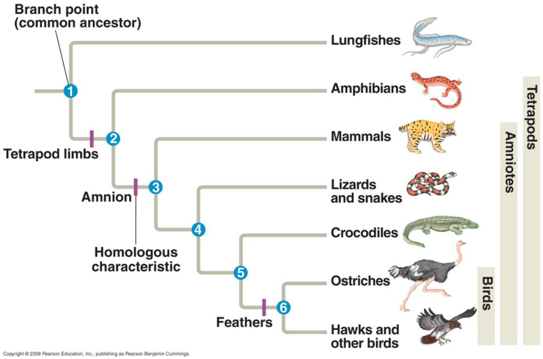
The Biological Problem

Basic Question in Biology

What properties are shared among organisms?

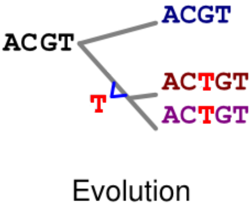
Homology: Organisms share Characteristics

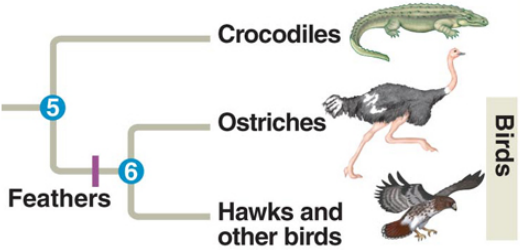
Descent from a common ancestor



http://www.bio.miami.edu/dana/160/160S13_5.html

Homology: Sequences match Positions





Sequence Similarity

Intuitively, similarity of two sequences refers to the degree of match between corresponding positions in sequence

Similarity between sequences

```
G G A T C G - - G A T T C G A A T G A T T C
| | | | | |     | | | |       | | | | | | |
G G A T C G C C T G C C - - - A T G A T T C
```

.

Similarity between strings

```
G A R F I E L D T H E L A S T F A - T C A T
| | | | | | | | | | | | | | | | |   | | | |
G A R F I E L D T H E V E R Y F A S T C A T
```

Similarity vs. Homology

- Similarity does not imply homology
- Similarity can occur by chance

But, what is expected...

But, homology is expected to cause similarity

Homology and Evolution

Homology is more difficult to detect over greater evolutionary

```
#mutations
0:            agtgtccgttaagtgcgttc
8:            agtgtccgcttcaaggggcgt
64:           acagtccgttcgggctattg
256:          cacgagtaagatatagct
1024:         acccttatctacttcctggagtt
2048:         agcgacctgcccaa
4096:         caaac
```

Sequence Alignment

Alignment specifies which positions in two sequences **match**

```
G G A T C G - - G A T T C G A A T G A T T C
| | | | | |     | | | |       | | | | | | |
G G A T C G C C T G C C - - - A T G A T T C
```

Edit Operations

```
G G A T C G - - G A T T C G A A T G A T T C
| | | | | |     | | | |       | | | | | | |
G G A T C G C C T G C C - - - A T G A T T C
```

- **13 matches**: Points where a single base do not change
- **4 mistmatches**: substitution (point mutation) of a single base
- **5 indels**: insertion or deletion of a base with respect to the ancestor sequence
  - 2 insertions (🟨)
  - 3 deletions (🟩)

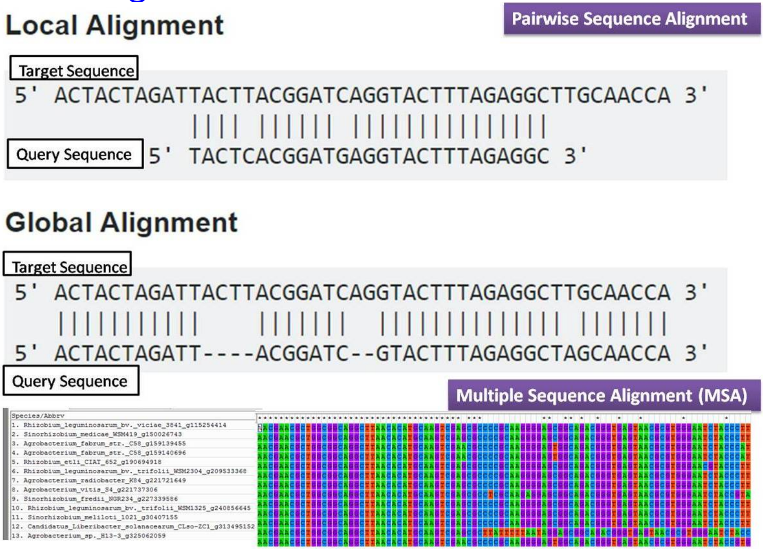
Sequence Alignment Problems

- What sorts of alignments should be considered?
- How to score alignments?
- How to find optimal or good scoring alignments?
- How to evaluate the statistical significance of scores?

First Question:

What sorts of alignments should be considered?

Types of Alignments



Pairwise Sequence Alignment

Local Alignment

Target Sequence

```
5' ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA 3'
             |||| |||||| |||||||||||||||
          5' TACTCACGGATGAGGTACTTTAGAGGC 3'
```

Query Sequence

Global Alignment

Target Sequence

```
5' ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA 3'
   |||||||||||    |||||||  |||||||||||||| |||||||
5' ACTACTAGATT----ACGGATC--GTACTTTAGAGGCTAGCAACCA 3'
```

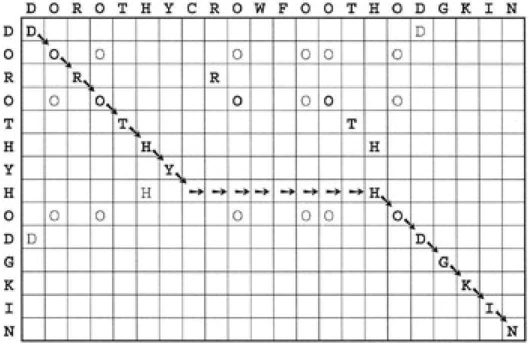
Query Sequence

Multiple Sequence Alignment (MSA)

```
Species/Abbrv
1. Rhizobium_leguminosarum_bv._viciae_3841_g115254414
2. Sinorhizobium_medicae_WSM419_g150026743
3. Agrobacterium_fabrum_str._C58_g159139455
4. Agrobacterium_fabrum_str._C58_g159140696
5. Rhizobium_etli_CIAT_652_g190694918
6. Rhizobium_leguminosarum_bv._trifolii_WSM2304_g209533368
7. Agrobacterium_radiobacter_K84_g221721649
8. Agrobacterium_vitis_S4_g221737306
9. Sinorhizobium_fredii_NGR234_g227339586
10. Rhizobium_leguminosarum_bv._trifolii_WSM1325_g240856645
11. Sinorhizobium_meliloti_1021_g30407155
12. Candidatus_Liberibacter_solanacearum_CLso-ZC1_g313495152
13. Agrobacterium_sp._H13-3_g325062059
```

Dot Plot Matrix: Strings

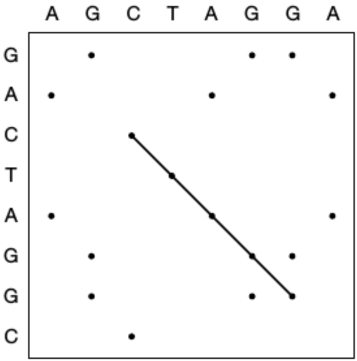
String A: DOROTHYCROWFOOTHODGKIN
String B: DOROTHYHODGKIN



	D	O	R	O	T	H	Y	C	R	O	W	F	O	O	T	H	O	D	G	K	I	N
D	D																	D				
O		O		O						O			O	O			O					
R			R						R													
O		O		O						O			O	O			O					
T					T										T							
H						H										H						
Y							Y															
H						H										H						
O		O		O						O			O	O			O					
D	D																	D				
G																			G			
K																				K		
I																					I	
N																						N

Dot Plot Matrix: Pair of Sequences

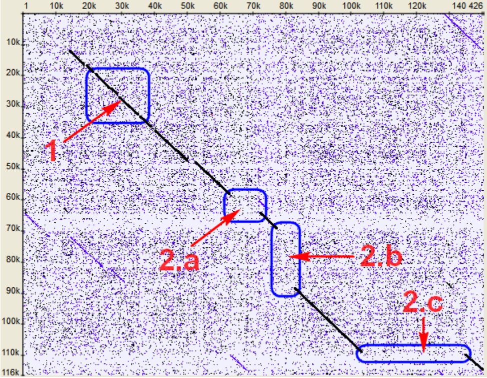
- Comparing two sequences:
  - AGCTAGGA
  - GACTAGGC
- Dots represent similarities between segments
- Diagonal of dots reveals similar elements



Not technically an ¨alignment¨ but it gives a picture of correspondence between pairs of sequences

Dot Plot Matrix: Interpretation

- **1 Matches:** looks like diagonals (continuous match or repeat)
- **2a Mutations:** gaps in the diagonal
- **2b Insertions:** gaps which lie only one axis (Y axis)
- **2c Deletions:** gaps which lie only one axis (X axis)



Dot Plot Matrix: Example

One alignment:

```
T C G G A T T C G T
| | | | | | | |
T C G C G T T C - -
```

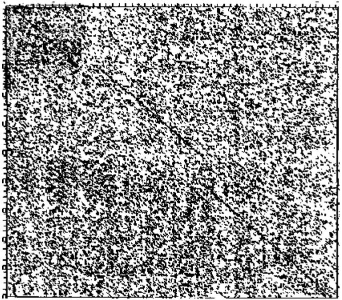
Alternate alignment:

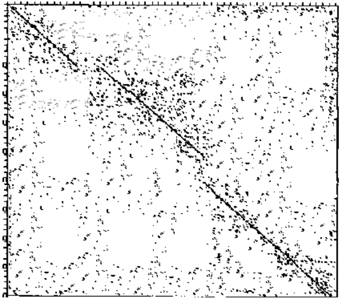
```
T C G G A T T C G T - -
| | |         | | |
T C G - - - - C G T T C
```

	T	C	G	G	A	T	T	C	G	T
T	●									
C		●								
G			●							
C								●		
G									●	
T						●				●
T							●			
C								●		

Dot Plot Limitations

- Problems with larger sequences sharing extensive regions of similarity
- Solution: filtering using a window size and threshold





Alignment Considerations:

What sorts of alignments should be considered?

- An alignment between two sequences is simply a pairwise match between the characters of each sequence.
- A true alignment (nucleotides or amino acids) reflects the evolutionary relationship between two or more homologous.
- Homology is not a matter of degree –at any given position in an alignment, sequences (and individual positions) either share a common ancestor or they do not.

In contrast, the overall similarity between two sequences can be described as a fractional value.

Second Question

How to score alignments?

Simple Alignments

Three possible **simple** alignments for AATCTATA y AAGATA:

```
AATCTATA            AATCTATA            AATCTATA
AAGATA               AAGATA               AAGATA
```

Three kinds of changes can occur:

1. **A mutation** replacing one character with another
2. **An insertion** adding one or more position
3. **A deletion** deleting one or more position

Scoring Simple Alignments

- Scoring Penalty Example: $\begin{cases} 1 \text{ for a match} \\ 0 \text{ for a mismatch} \end{cases}$
- Scoring the Alignments:

```
AATCTATA             AATCTATA           AATCTATA
AAGATA                AAGATA              AAGATA
--------             --------           --------
Score = 4            Score = 1          Score = 3
```

Scoring function for a Simple Alignment:

$$\sum_{i=1}^{n} \begin{cases} \text{match score if seq1=seq2} \\ \text{mismatch score if } seq1 \neq seq2 \end{cases}$$

Alignment with Gaps

- Insertions and deletions events complicates sequence alignments
- The number of possible alignments increase vastly

$$\binom{3}{7} = 28$$

Only 5 of the 28 possible alignments :

```
AATCTATA    AATCTATA    AATCTATA    AATCTATA    AATCTATA
AAG-AT-A    AA-G-ATA    AA--GATA    A-A-GATA    AA-GAT-A
```

Scoring Alignments with Gaps

- Scoring Penalty Example: $\begin{cases} \text{-1 for gaps} \\ +1 \text{ for a match} \\ 0 \text{ for a match} \end{cases}$
- Scoring the Alignments:

```
AATCTATA  AATCTATA  AATCTATA  AATCTATA  AATCTATA
AAG-AT-A  AA-G-ATA  AA--GATA  A-A-GATA  AA-GAT-A
--------  --------  --------  --------  --------
Score = 1 Score = 3 Score = 3 Score = 2 Score = 2
```

Scoring function for a Simple Alignment:

$$\sum_{i=1}^{n} \begin{cases} \text{gap penalty, if seq1=''-'' or seq2=''-''} \\ \text{match score, if seq1=seq2} \\ \text{mismatch score, if } seq1 \neq seq2 \end{cases}$$

Origination and Length Penalties

- **Indel events (indels)**: Insertion and Deletion Events

What is more likely from an evolutionary perspective?

35

```
Without gaps        Multiple indels     Few indels
----------------    ----------------    ----------------
AATCTATAGGGTAGAT    AATCTATAGGGTAGAT    AATCTATAGGGTAGAT
  AAGATAGTAA        AA-G-AT-A-GT--AT    AAG--ATAG--TA--T
```

- Extended are more frequent than single multiple **indels events**
- Scoring function biased to reward alignments **extending gaps**

Scoring Alignments with Gap Penalty

- Scoring Penalty Example: $\begin{cases} \text{-2 for origination penalty} \\ \text{-1 for length penalty} \\ +1 \text{ for a match} \\ \ 0 \text{ for a match} \end{cases}$

```
AATCTATAGGGTAGAT        AATCTATAGGGTAGAT
AA-G-AT-A-GT--AT        AAG--ATAG--TA--T
Score = -3              Score = 0
```

Scoring function for a Simple Alignment:

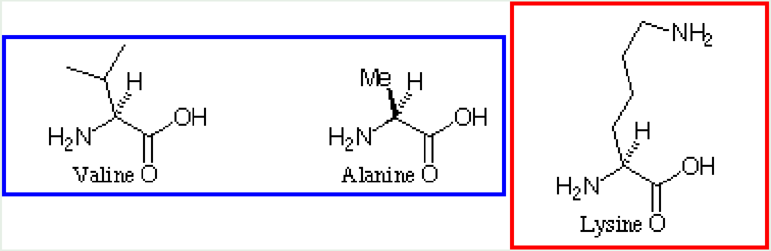
$$\sum_{i=1}^{n} \begin{cases} \text{origination gap penalty, if seq1="-" or seq2="-"} \\ \text{length gap penalty, if seq1="-" or seq2="-"} \\ \text{match score, if seq1=seq2} \\ \text{mismatch score, if } seq1 \neq seq2 \end{cases}$$

Scoring Matrices:

Taking account conservative substitutions

- Some substitutions are more common than others.

Proteins: alanine substituted for valine instead lysine



Mismatch penalty can be broken down as gap penalty

Scoring Matrices: DNA Sequences

Identity Matrix

	A	T	C	G
A	1	0	0	0
T	0	1	0	0
C	0	0	1	0
G	0	0	0	1

BLAST Matrix

	A	T	C	G
A	5	-4	-4	-4
T	-4	5	-4	-4
C	-4	-4	5	-4
G	-4	-4	-4	5

Transition Transversion

	A	T	C	G
A	5	-4	-4	-4
T	-4	5	-4	-4
C	-4	-4	5	-4
G	-4	-4	-4	5

- Scoring matrix is used to score each nongap position
- Transitions transversion matrix provides mild penalty for transitions:
  - Purine (A or G) is replaced with another purine
  - Pyramidine (C or T) is replaced with another purine

Scoring Matrices: Amino Acid sequences

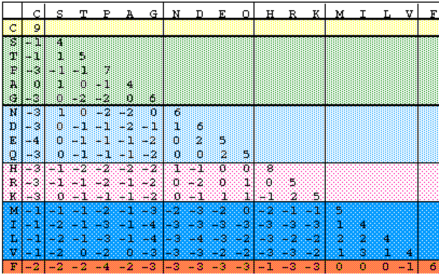
PAM (Point Accepted Mutation):

- Computed by observing substitution rates
- Used to score closely related sequences

BLOSUM (BLOcks SUbstitution Matrix):

- Computed by clustering ungapped alignments
- Used to score more distant related sequences

Blosum



	C	S	T	P	A	G	N	D	E	O	H	R	K	M	I	L	V	F
C	9																	
S	-1	4																
T	-1	1	5															
P	-3	-1	-1	7														
A	0	1	0	-1	4													
G	-3	0	-2	-2	0	6												
N	-3	1	0	-2	-2	0	6											
D	-3	0	-1	-1	-2	-1	1	6										
E	-4	0	-1	-1	-1	-2	0	2	5									
Q	-3	0	-1	-1	-1	-2	0	0	2	5								
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8							
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5						
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5					
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5				
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4			
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4		
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4	
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6

Third Question:

How to find optimal or good scoring alignments?

Exhaustive search

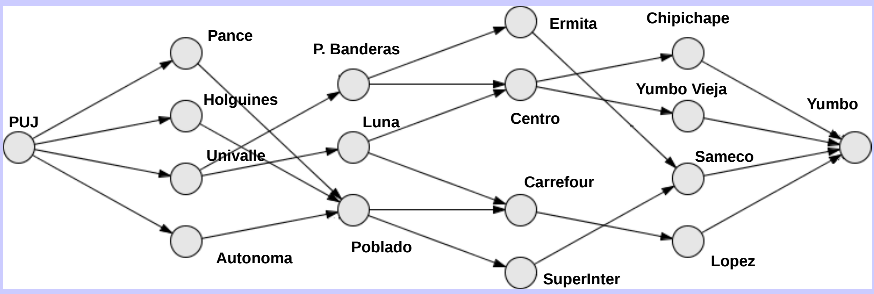
- Exhaustive search is not feasible for most sequences
- Two modest-sized sequences of 100 and 95 nucleotides
    - ˜75 million possible alignments
- For larger sequences, search becomes **intractable**

Impossible to compute in a reasonble amount of time

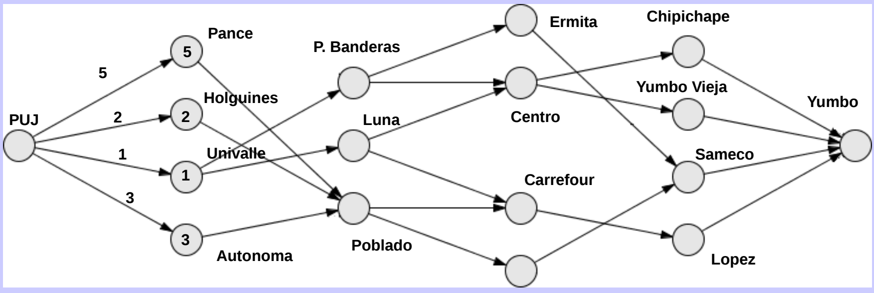
Dynamic Programming

- A method of breaking a problem apart into reasonably sized problems and using these partial results to compute the final answer.

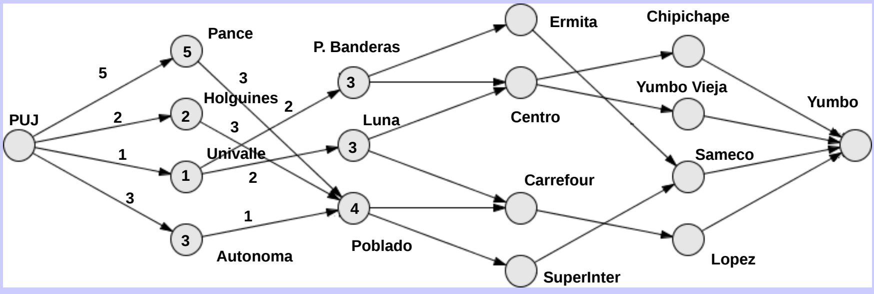
Example: Shortest Path Problem (Initial)



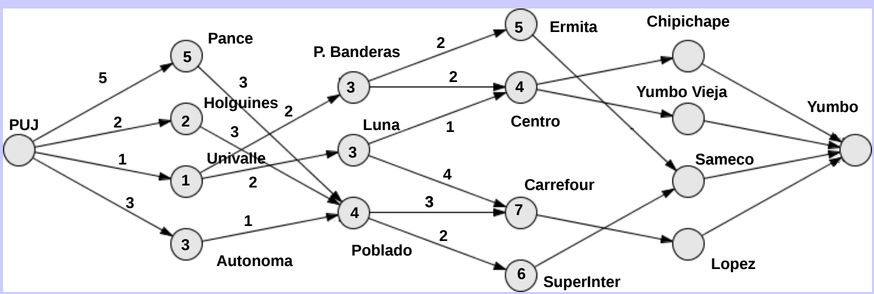
Example: Shortest Path Problem (01)



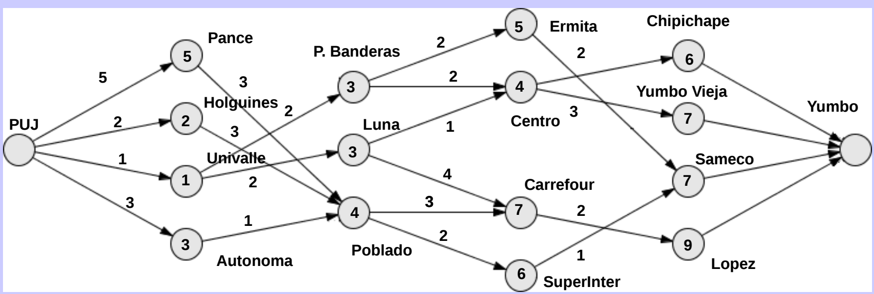
Example: Shortest Path Problem (02)



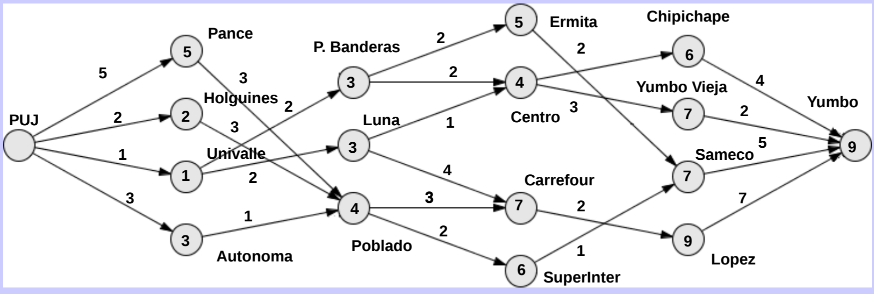
Example: Shortest Path Problem (03)



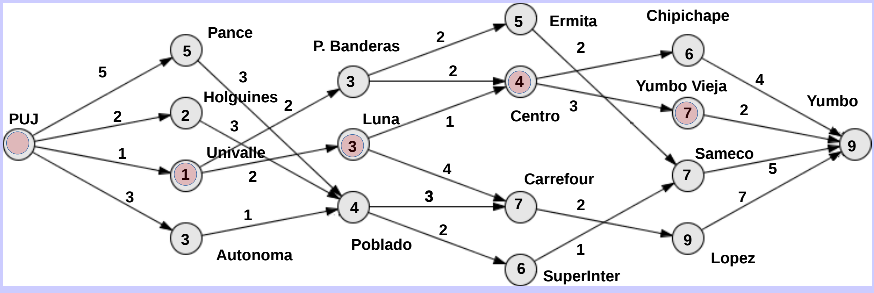
Example: Shortest Path Problem (04)



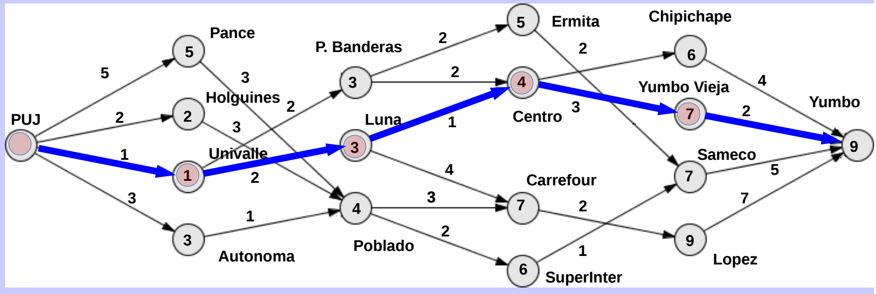
Example: Shortest Path Problem (Final)



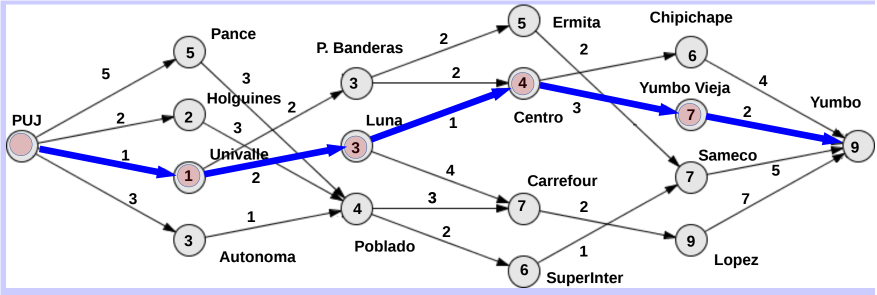
Example: Shortest Path Problem (Backtracking)



Example: Shortest Path Problem (Shortest Path)



Example: Shortest Path Problem (Recursive Algorithm)



```
ShortestPath (PUJ, Yumbo):
   min(  5 + ShortestPath (Pance, Yumbo);
         2 + ShortestPath (Holgines, Yumbo);
         1 + ShortestPath (Univalle, Yumbo);
         2 + ShortestPath (Autonoma, Yumbo)
       )
```

Too Many Recursive Calls

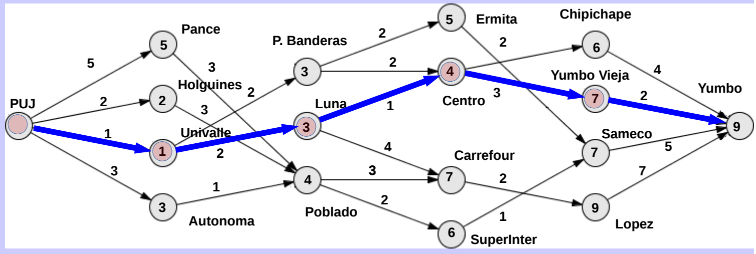
```
ShortestPath (Pance, Yumbo):
   min(  5 + ShortestPath (Poblado, Yumbo))
ShortestPath (Poblado, Yumbo):
   min(  5 + ShortestPath (Carrefour, Yumbo);
         5 + ShortestPath (SuperInter, Yumbo))
...
ShortestPath (Sameco, Yumbo):
   min(5)

5
```

```
ShortestPath (Univalle, Yumbo):
   min(  2 + ShortestPath (PBanderas, Yumbo);
         2 + ShortestPath (Luna, Yumbo))
ShortestPath (PBanderas, Yumbo):
   min(  2 + ShortestPath (Ermita, Yumbo);
         2 + ShortestPath (Centro, Yumbo);)
...
```

```
ShortestPath (Carrefour, Yumbo):
...
```

```
ShortestPath (SuperInter, Yumbo):
...
```

Example: Shortest Path Problem (Dynamic Programming)



	Pance	Pobl
PUJ	5	
Pance		8
Holg.		5
Univ		1
Auton		4

→

	Pance	Pobl
PUJ	5	4
Pance		8
Holguines		5
Univ		1
Auton		4

→

	Pance	Pobl	Luna	PBan	Centro
PUJ	5	4			4
Pance		8			
Holg		5			
Univ		1	3	3	
Auton		4			
Centro			1	2	

Needleman and Wunsch Algorithm

- Needleman and Wunsch were the first to apply DP to sequence alignments

Key to understanding DM approch to sequence alignment

Observing how the alignment problem is broken down into subproblems

Example: Align the sequences CACGA y CGA

```
                 *
CACGA        C | ACGA       − |CACGA      C | ACGA
CGA          C | GA         C | GA        − |CGA
-----        ------         -------       ------
0            +1             −1            −1
```

```
                     *
ACGA       A | CGA     − |ACGA      A | CGA
GA         G | A       G | A        − |GA
----       -----       ------       ------
+1         +1          −1           −1
```

```
                *
ACGA       A | CGA     − |ACGA     A | CGA
A          A |         A               − |A
----       -----       ------      -----
+1−1       +1          −1          −1
```

```
CGA                      Result    C − A C G A
                                   C G A
------                             -----------
                         Score     +1−1+1          =    1
```

Dynamic Programming Matrix

```
Sequence 1: CACGA
Sequence 2: CGA
```

Sequences CACGA y CGA

	–	C	A	C	G	A
–						
C						
G						
A						

Dynamic Programming Matrix:

Initialization with Penalty Gaps

```
Sequence 1: CACGA
Sequence 2: CGA
```

- Uniform Penalty Gap of -1

Moves:

- Horizontal: gap in the X-Axis
- Vertical: gap in the Y-Axis
- Diagonal: match or mismatch

CACGA y CGA

	–	C	A	C	G	A
–	0	-1	-2	-3	-4	-5
C	-1					
G	-2					
A	-3					

Dynamic Programming Matrix:

Edit Operations and Scoring Function

Edit Operations

```
Sequence:                CACGA
----------------------------
Substitution:            GACGA
Indel:
    Deletion  (Del):     -ACGA
    Insertion (Ins):     TGAGA
```

Scoring Function

```
Match:      +1
Mismach:     0
Indel:      -1
```

```
Sequence 1: CACGA
Sequence 2: CGA
```

CACGA y CGA

	–	C	A	C	G	A
–	0	-1	-2	-3	-4	-5
C	-1					
G	-2					
A	-3					

Dynamic Programming Matrix:

For each step

Scoring Function

```
Match:      +1
Mismach:     0
Indel:      -1
```

Compute the max score for each cell:

- According to the score
- According to the neighbors

Pos		1	2	3	4	5	6
		–	**C**	**A**	**C**	**G**	**A**
1	–	0	-1	-2	-3	-4	-5
2	**C**	-1	↘ ↓ +1 -1 → -1 **1**				
3	**G**	-2					
4	**A**	-3					

Dynamic Programming Matrix:

Finish with backtracking

Scoring Function

Match:	+1
Mismach:	0
Indel:	-1

CACGA	→	CACGA
CGA	→	C--GA

	–	C	A	C	G	A
–	0	-1	-2	-3	-4	-5
C	-1	↖ **1**	← **0**	← **-1**	-2	-3
G	-2	0	1	0	↖ **0**	-1
A	-3	-1	1	1	0	↖ **1**

Dynamic Programming Matrix:

Exercise

ACTCG → ?

ACAGTAG → ?

	–	A	C	T	C	G
–						
A						
C						
A						
G						
T						
A						
G						

Scoring Function

```
Match:      +1
Mismach:     0
Indel:      -1
```

Dynamic Programming Matrix:

Solution

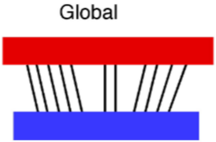
Scoring Function

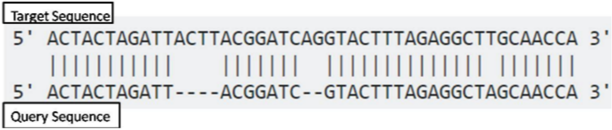
```
Match:      +1
Mismach:     0
Indel:      -1
```

```
ACTCG      →    AC--TCG
ACAGTAG    →    ACAGTAG
```

	–	A	C	T	C	G
–		-1	-2	-3	-4	-5
A	-1	1	0	-1	-2	-3
C	-2	0	2	1	0	-1
A	-3	-1	1	2	1	0
G	-4	-2	0	1	2	2
T	-5	-3	-1	1	1	2
A	-6	-4	-2	0	1	1
G	-7	-5	-3	-1	0	2

Global vs Local Alignments

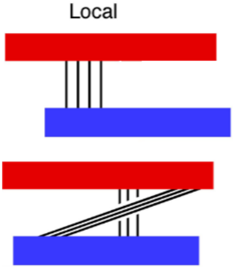


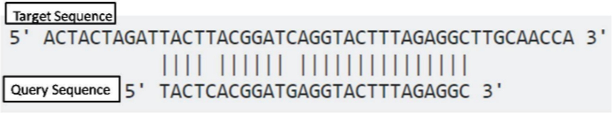


Target Sequence

```
5'  ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA  3'
    |||||||||||    |||||||  |||||||||||||| |||||||
5'  ACTACTAGATT----ACGGATC--GTACTTTAGAGGCTAGCAACCA  3'
```

Query Sequence





Target Sequence

```
5'  ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA  3'
              |||| |||||| |||||||||||||||
         5'   TACTCACGGATGAGGTACTTTAGAGGC  3'
```

Query Sequence

The Smith-Waterman Algorithm

Restrictions of the Global Alignments

- Not restricted to align the entire sequence
- Not restricted to have negative partial scores
- Not restricted to start the traceback in the botton-right

The Smith-Waterman Algorithm

Modifications to the Global Alignment Algorithm

- First row and first column of the DP matrix are filled with 0s
- Allow preceding and trailing indels without penalty:
  - if a score < 0 then put 0
- Look the highest-scoring path starting in the cell with the highest-score.

The Smith-Waterman Algorithm

An example

Scoring Function

```
Match:      +1
Mismach:    -1
Indel:      -1
```

```
CACGAT    →    CACGAT
CGAA      →      CGA
```

	–	C	A	C	G	A	T
–	0	0	0	0	0	0	0
C	0	1	0	↖**1**	0	0	0
G	0	0	0	0	↖**2**	1	0
A	0	0	0	0	1	↖**3**	2
A	0	1	0	0	0	2	2

The Smith-Waterman Algorithm

An excercise

AACCTATAGCT → ?
GCGATATA → ?

Scoring Function

```
Match:      +1
Mismach:    -1
Indel:      -1
```

	–	A	A	C	C	T	A	T	A	G	C	T
–												
G												
C												
G												
A												
T												
A												
T												
A												

The Smith-Waterman Algorithm

The solution

Scoring Function

Match:	+1
Mismach:	-1
Indel:	-1

AACCTATAGCT → aaccTATAgct
GCGATATA → TATA

	–	A	A	C	C	T	A	T	A	G	C	T
–	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	1	0	0
C	0	0	0	1	1	0	0	0	0	0	2	0
G	0	0	0	0	0	0	0	0	0	1	1	0
A	0	1	1	0	0	0	1	0	1	0	0	0
T	0	0	0	0	0	↖ **1**	0	0	0	0	0	0
A	0	1	1	0	0	0	↖ **2**	1	0	0	0	0
T	0	0	0	0	0	1	1	↖ **3**	2	0	0	0
A	0	1	1	0	0	0	1	2	↖ **4**	0	0	0

Assignments

1. Reading ¨Significance of Alignments¨. Introduction to Bioinformatics¨ (Arthur M. Lesk) p.267.
2. Reading ¨Databases Searches¨. Fundamental Concepts of Bioinformatics (Krane & Raymer) p.48.