trivial
Hamming Distance: d_H
Number of mismatches.

Median String: $|\Sigma|_c$ alphabet len
AAA..A→TT..T
find best hamming distance

$m = |p|$
$n = |t|$
Brute force: $O(m \cdot n)$
KMP → $O(m+n)$

KMP Match:

```
F ← failureFunction(P)
i ← 0
j ← 0
while i<n
    if T[i] = P[j]
        if j=m-1
            return i-j
        else
            i←i+1
            j←j+1
    else
        if j>0
            j← F[j-1]
        else
            i←i+1
            j←0
return -1
```

KMP failure:

```
F[0]←0
i←1
j←0
m←len(P)
while i<m
    if P[i]=P[j]
        F[i]←j+1
        i←i+1
        j←j+1
    else if j>0
        j←F[j-1]
    else
        F[i]←0
        i←i+1
```

Suffix Trees:
Collapsed keyword trees:

Let s=abab, suffix tree of s is a compressed tree of all suffixes of s=abab$ which are
$
b$
ab$
bab$
abab$
} put these into tree

~~(m) build~~ find(P) O(n+m) to find

Weiner Algorithm can do it in linear time!

Boyer Moore:
Bad Char:
Last occurence of each char.
Good Suffix 1:
if t∈P matched with text, find last occurance of T left side.
Good Suffix 2:

find t' left side.

Fingerprint: ~~Q(n)~~
$f(P) \Rightarrow O(m)$] $h = f \bmod q$
Rabin Karp: $O(n)$
Hash pattern. Hash text's first m chars and compare 1-m, 2-m+1 ... if same, compare strings.

Select $q > m$, q → prime

Finite Automata: linear $O(n)$ good
States: 0,1,2...,m
preprocess → $O(m|\Sigma|)$
Memory ↗ bad

Bitap Algorithm: Shift-And [$O(mn)$]
$M(i,j) = 1$ iff $P[1...i] = T[j-i+1...j]$
m rows
n cols } M
Bitshift(a) → shift a to right by adding 1 to first bit
a001 | 0b1000 0000

$M(j) = \text{BitShift}(j-1) \wedge U(T(j))$
Define U for each char in alphabet.
if last row has 1: found match

Pattern Matching: $O(nm)$
Suffix Trees: $O(n)$ Let N total n's
Multiple Pattern Match → brute force ([$O(kmn)$])
keyword tree → $O(N)$ + Naive thread → ($O(N+nm)$)
or
+ Aho-Corasick → ($O(N+m)$)
↓
Construct using patterns and thread over text

Aho Corasick: Search in keyword trees!
Failure links:
$L(v)$ → the word that is the concatnetion of the chars until node v from 0.
$lp(v)$ → longest suffix of $L(v)$ which is a prefix of some word starting from "0" node
failure link
→ $O(n)$ failure links

Burrows-Wheeler Transformation
bzip2
Generate all rotations of word
Sort them. $ < all alphabet
Output last column!
Last-to-first char map
n^{th} "x" char on left is also n^{th} "x" on right!

Occ(j,'c') ⇒ number of occurences of char 'c' until jth pos. (inclusive) (table), cnt('c') table contains occurence numbers of each char!, rank('c') first occurence of each char in first word

Edit Distance: d_E [non-trivial]
Minimum number of elementary operations to get S_2 from S_1.

Multiple Aligament:
Frequencies for each column
p_A, p_G, p_C, p_T, p_- (gap) char col
$p_x = \frac{\text{\# of A's}}{\text{\# of rows}} \rightarrow p(y,j)$

$S(x,j) = \sum_y (\delta(x,y) \times p(y,j))$
char on sequence, col

char on sequence	Profile								
	0	-1	-2	-3	-4	-5	-6	-7	-8
C	-1	0,8	-0,2	-1,2	-2,2	-2	-3	-4	-5
A	-2	-0,2	2,8	1,8	0,8	-0,2	-1,2	-1	-2
G	-3	-1,2	1,8	4,8					-0,2
G	-4	-2,2	0,8		4,4	3,4			0,4
T	-5	-3,2	-0,2				3,4		3,4
A	-6	-4,2	-1,2					7,4	6,7
C	-7	-5,2	-2,2						2,6

Scoring Multiple Alignments:
Multiple LCS: Match number only
AAA → match!
AAA
AAT
ATC

Entropy:
$-\sum p_x \log_2 p_x$
$x = A, T, G, C$
if $p_x = 0$, then 0

A	1	0,75	0,5
T		0,25	0,25
G			
C			0,25

worst entropy: A T G C 0,25 for all! $-\frac{1}{4}\log_2\left(\frac{1}{4}\right)4 = \boxed{+2}$
Sum all entropies to find alignment score (of cols)

Use gap-[illegible] while openning gaps! $O(nm)$ to fill

Global Alignment (Pairwise)

$$s_{i,j} = \max \begin{cases} s_{i-1,j-1} + \text{match\_score} & \text{if } v_i = w_j \\ s_{i-1,j-1} + \text{mismatch\_sc} & \text{if } v_i \neq w_j \\ s_{i-1,j} + \text{gap\_score} \\ s_{i,j-1} + \text{gap\_score} \end{cases}$$

$$\text{Sequence Identity} = \frac{\text{num\_matches}}{\text{len(alignment)}} \times 100$$

Local Alignment:
Dont ~~decrease~~ scores under '0'

$$s_{i,j} = \max \begin{cases} s_{i-1,j-1} + \delta(v_i, w_j) \\ s_{i-1,j} + \delta(v_i, -) \\ s_{i,j-1} + \delta(-, w_j) \\ 0 \end{cases}$$

$$V(i,j) = \max \begin{cases} V(i-1,j-1) + S(s_1(i), j) \\ V(i-1,j) + \delta(s_1(i), '-') \\ V(i,j-1) + \delta('-', j) \end{cases}$$

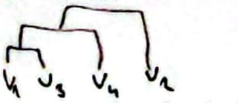
s_1 → sequence! not score!

A		1					1	
C	0,6			0,2	1			0,4
G			1					
T	0,2					1		0,6
-	0,2			0,8				

ClustalW
similarity = frequency of exact matches
Guide Tree ex..

	v_1	v_2	v_3	v_4
v_1	-			
v_2	.17	-		
v_3	.87	.28	-	
v_4	.59	.33	.62	-

then



align 1-3, then 4, then 2 with gaps and mismatches

Sum of Pairs Score
Calculate as pairwise score! with gaps and mismatches for all pair:
if 4 alignment then 6 probabilities

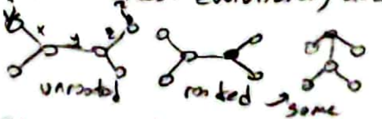
UPGMA:
$x_i \rightarrow C_i$ assignment (one for each x)
Find C_i and C_j such that d_{ij} is min.
Let $C_k \leftarrow C_i \cup C_j$
Add vertex connecting C_i, C_j height ← $\frac{d_{ij}}{2}$
delete C_i, C_j
repeat until 1 cluster remains.

Star Alignment:
Sim→score of pairs
Compute sim for every pair (i,j)
$\text{star\_score}(i) = \sum_{j \in P-(i)} \text{sim}(i,j)$
max(starscore) → center pattern
align according to it.

$d_{ic} + d_{jc} = D_{ij}$
$d_{ic} + d_{kc} = D_{ik}$
$+ \; d_{jc} + d_{kc} = D_{jk}$
using Math:
$d_{ic} = \frac{D_{ij} + D_{jk} + D_{ik}}{2}$
can be calculated.

n leave tree has 2n-3 edges.
if n>3 this may not be solvable!
not same!

Evolutionary Trees:
Leaves: existing species
Interval vertices: ancestors
Root: oldest evolutionary ancestor



Edges may have weight → # of mutations or time estimate of evolution process.
$d_{ij}(T)$ → tree distance between i and j
from 1 to 2: $d_{1,2}(T) = x+y+z$
D_{ij} → n×n distance matrice
edit

Degenerate Triples: (D
$D_{ij} + D_{jk} = D_{ik}$
j can be removed.
If there is no DT, create one by shortening edges.
↓
hanging (connected to a leaf

Change Problem:

$M = c_1 i_1 + c_2 i_2 + .$

where $i_1 + i_2 + ...$ minimized.

$c_1, c_2 ...$ denominateors.

Algorithm:

$$\text{minCoins}(M) = \min \begin{cases} \text{minCoins}(M-c_1)+1 \\ \vdots \\ \text{minCoins}(M-c_d)+1 \end{cases}$$

Additive_Phylogeny(D)

```
if D is a 2x2 matrix:
    T= tree of a single edge of len $D_{1,2}$
    return T
if D is non-degereate:
    $\delta$ = trimming parameter of matrix D
    for all $1 \le i \ne j \le n$
        $D_{ij} \Leftarrow D_{ij} - 2\delta$
else
    $\delta = 0$
    Find a triple i,j,k such that $D_{ij} + D_{jk} = D_{ik}$
    $x = D_{ij}$
    Remove jth row and jth column
    T= Additive_Phylogeny(D)
    Add a new vertex v to T at distence x from i to k
    Add j back to T by creating edge (v,j) of len(?)
    for every leaf l in T:
        if distance from l to v in the tree $\ne D_{l,j}$
            output "matrix is not additive
            return
    Extend all hanging edges by $\delta$
    return T
```

Four Point Condition:

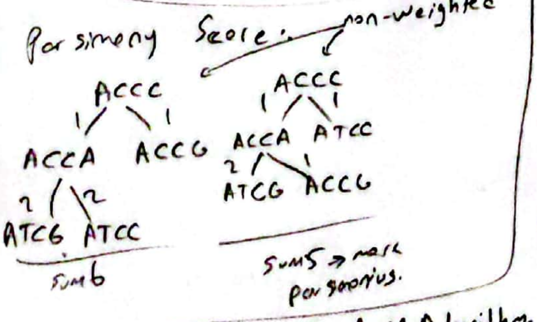
i,j,k,l → four leaves of tree

Two of $D_{x,y}$'s are equal and third sum is smaller, four point cond. satisfied.

Theorem: nxn matrix is additive iff four point cond. satisfied for all quartets!

Parsimony Score:

non-weighted



Weighted parsimony: Sankoff Algorithm.

Begin from leaf.

leaf have score according to the character it have.

After: $s_t(\text{parent}) = \min\{s_i(\text{left child}) + \delta_{i,t}\} + \min\{s_j(\text{right child}) + \delta_{j,t}\}$

Clustering:

UPGMA

K-Means Clustering:

```
Lloyd Algorithm:
    Arbitrarily assign K cluster centers.
    while cluster centers keep changing:
        Assign each data point to cluster center (nearest)
        Find center of gravity of each cluster
        and assign them as new cluster centers.
```

another

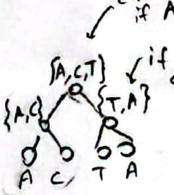
Greedy Algorithm:

```
Select arbitrary partition P into k clusters
while forever
    bestchange ← 0
    for every C (cluster)
        for i not in C (element)
            if moving i into C reduces score:
                if $\text{cost}(P) - \text{cost}\,P_{i \to C} > \text{best Change}$
                    bestchange ← $\text{cost}\,P - \text{cost}\,P_{i \to C}$
                    $i^* = I$
                    $C^* = C$
    if best Change > 0
        move $i^*$ to $C^*$
    else
        return P.
```

Graph Clustering:

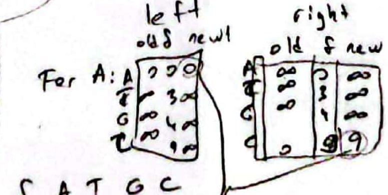
Clique Graph: A graph whose every vertex is connected to every other vertex.

Fitch Algorithm (Small parsimony)

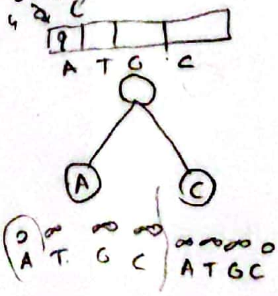


choose arbitrarily if A:

if it has parent's label choose it else choose arbitrarily.



δ	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0



Types of Rearrangements.

Reversal: 1 2 3 4 5 6 → 1 2 -5 -4 -3 6

Translocation: 1 2 3 / 4 5 6 → 1 2 6 4 / 5 3

Fusion: 1 2 3 4 / 5 6 → 1 2 3 4 5 / 6

← Fission

π: 1 2 3 4 5 6 7 8

$\rho(3,5)$ ↓

1 2 5 4 3 6 7 8

$\rho(5,6)$ ↓

1 2 5 4 6 3 7 8

if π = 1 2 3, 6 5 4

already arranged,

prefix(π) = 3

increase it every step.

flip 6 5 (two by two

CAST: S G θ

P = ∅

while S ≠ ∅

Distance Graph:

Threshold θ.

if distance < θ connect vertices.

```
Break Point Reversal Search($\pi$)
while $b(\pi) > 0$
choose $\rho$ min($b(\pi \cdot \rho)$)
```

if $\pi_{i+1} = \pi_i \pm 1$

π_i and π_{i+1} are adjacent.

there is a breakpoint between any adjacent that are not consecutive.

BLAT:

Index database and find query but BLAST index query and search database

Blast: O(nm)

Dict: All words of len(w)

no gap, score > threshold returns.

Gapped blast

Pattern Hunter: Spaced seed:

If blast w=11 but in Pattern hunter

11111111 ...

11101001 ......

18 elements there are 11, 1's

Group of Models

Extend each hit.