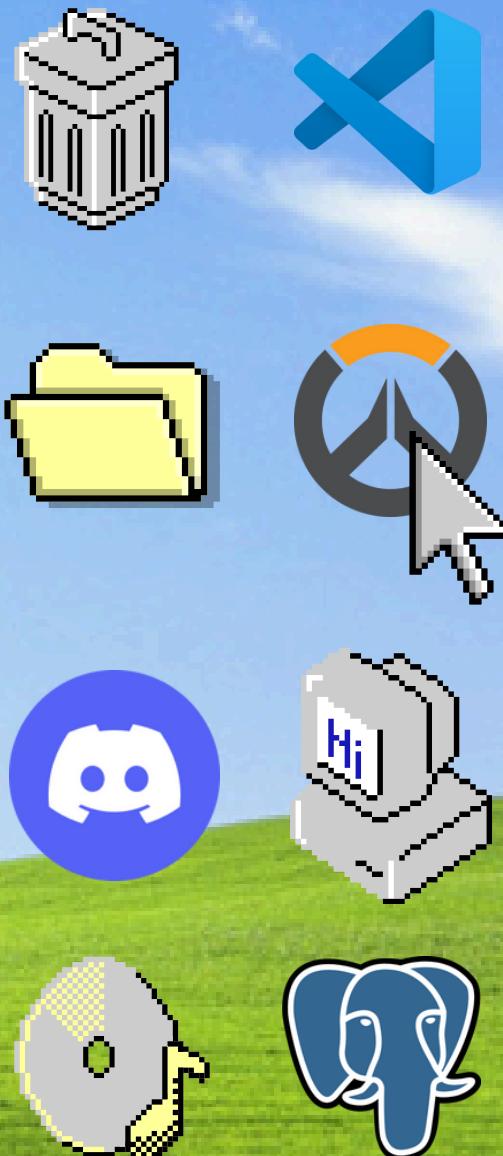
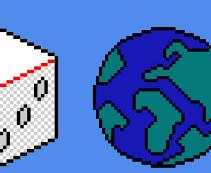


CRACKING DE CONTRASEÑAS

Mediante diferentes métodos

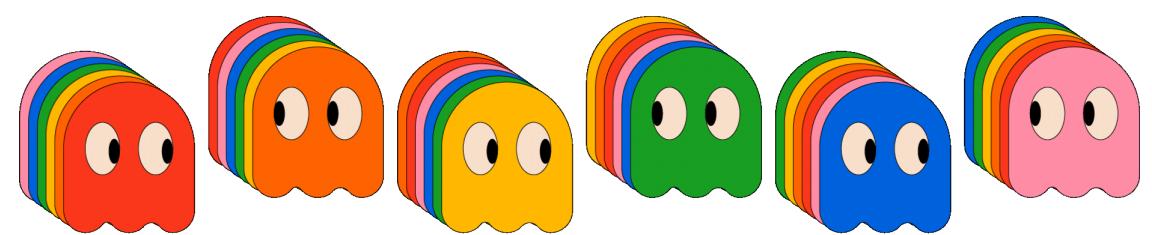


Eduardo Ramos Ochoa
304489918
Andres Santiago Aguirre
Macias
220293594
Cesar Emmanuel Gómez
Martínez 214843698



Contexto del problema

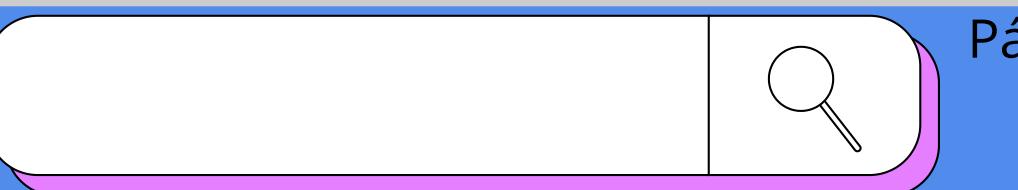
La suma de los distintos enfoques algorítmicos, en concreto, al algoritmo de “cracking de contraseñas”, partiendo desde, fuerza bruta y, continuando con divide y vencerás, programación dinámica, concluyendo con enfoque voraz, todo, en el mismo orden mencionado.

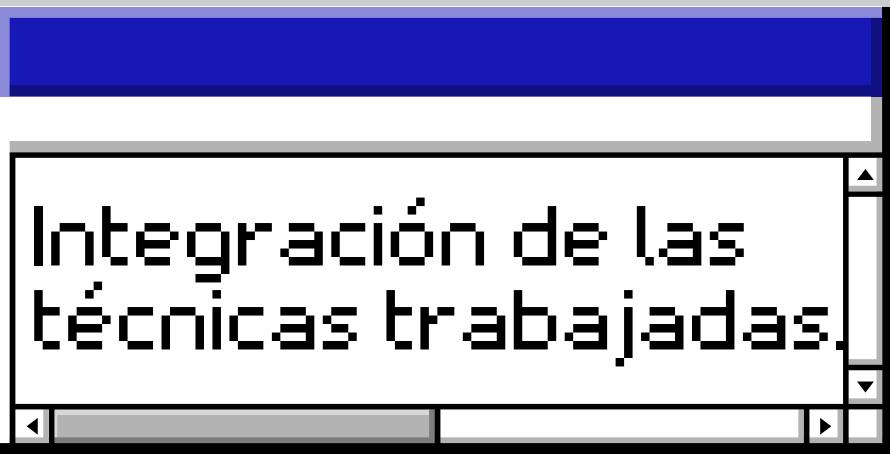


Algoritmo base.

Cracking de contraseñas

Utilizada para descifrar contraseñas al probar sistemáticamente cada combinación posible de caracteres hasta dar con la correcta.
Su velocidad depende críticamente de la complejidad de la contraseña.





Cracking Modes - Demo académico (v2)

Target (plain o hash): lemmanruss
(si pones texto plano, selecciona 'plain' checkbox)

Hash: sha256

Modo: Fuerza Bruta (FB) DyV (Divide & Vencerás) Prog Dinámica (edits desde wordlist) Voraz (Prim + MST)

Wordlist: C:/Users/PCREATIORS/Desktop/5TO SEMESTRE CUCEI ICOM/ANALISIS DE ALGORITMOS/EXPO CODIGO P\

Alphabet: abcdefghijklmnopqrstuvwxyz0123456789

Max len (FB/DyV): 4 PD: max edits: 2
6 Prim: top N start: 3
Random pwd len: 6

Candidate limit (global, 0 = no limit): 0

Generar contraseña aleatoria (oculta)

Start Stop Cargar wordlist (preview)

```
[FB] checked 1650000 candidates (last: 8mel)
[FB] checked 1655000 candidates (last: 8p9h)
[FB] checked 1660000 candidates (last: 8t4d)
[FB] checked 1665000 candidates (last: 8xy9)
[FB] checked 1670000 candidates (last: 8lt5)
[FB] checked 1675000 candidates (last: 85o1)
[FB] checked 1680000 candidates (last: 89jx)
[FB] checked 1685000 candidates (last: 9det)
[FB] checked 1690000 candidates (last: 9g9p)
[FB] checked 1695000 candidates (last: 9k4l)
[FB] checked 1700000 candidates (last: 9ozh)
[FB] checked 1705000 candidates (last: 9sud)
[FB] checked 1710000 candidates (last: 9wo9)
[FB] checked 1715000 candidates (last: 90j5)
[FB] checked 1720000 candidates (last: 94e1)
[FB] checked 1725000 candidates (last: 979x)
[RESULT] No encontrado (method=FB) time=1.3741s checks=1727604
[STATS] Time total: 1.3746s | Algorithm time: 1.3741s | Checks: 1727604 | Time/check: 7.953894e-07s
[THEORY] Temporal: O(k^n) | Espacial: O(1) (o O(n) para candidato)
```

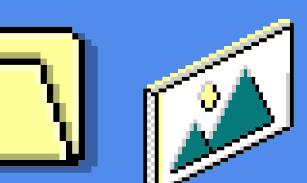
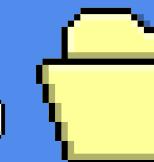
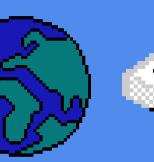
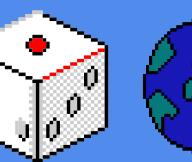
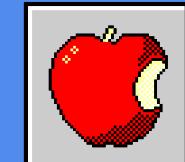
Tiempos

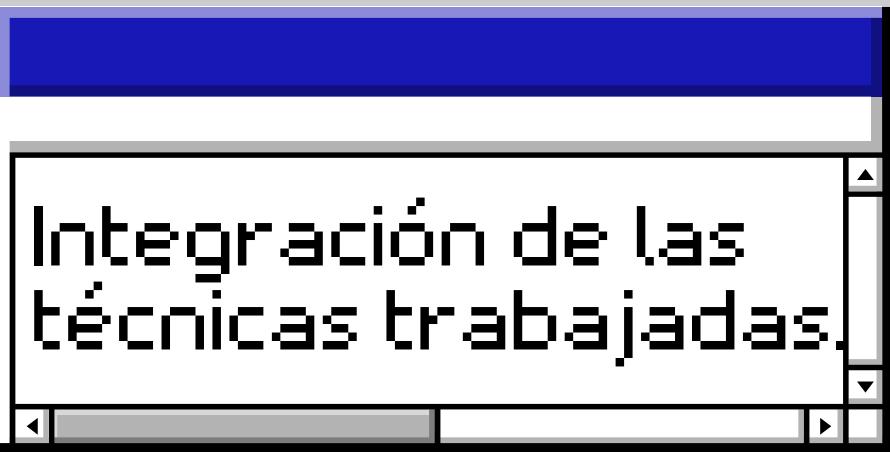
1.3741 segundos

Fuerza Bruta

En lugar de probar todas las combinaciones, el atacante utiliza una lista predefinida de palabras o frases comunes (un "diccionario").

Start





Cracking Modes - Demo académico (v2)

Target (plain o hash): lemmannruss plain
(si pones texto plano, selecciona 'plain' checkbox)

Hash: sha256

Modo: DyV (Divide & Vencerás) Prog Dinámica (edits desde wordlist) Voraz (Prim + MST)

Wordlist: C:/Users/PCREATIORS/Desktop/5TO SEMESTRE CUCEI ICOM/ANALISIS DE ALGORITMOS/EXPO CODIGO P/

Browse

Alphabet: abcdefghijklmnopqrstuvwxyz0123456789

Max len (FB/DyV): 4 PD: max edits: 2
6 Prim: top N start: 3
Random pwd len: 6

Candidate limit (global, 0 = no limit): 0

Generar contraseña aleatoria (oculta)

Start Stop Cargar wordlist (preview)

```
[DyV] checked 1655000 candidates (len=4, last=8p9h)
[DyV] checked 1660000 candidates (len=4, last=8t4d)
[DyV] checked 1665000 candidates (len=4, last=8xy9)
[DyV] checked 1670000 candidates (len=4, last=8lt5)
[DyV] checked 1675000 candidates (len=4, last=85o1)
[DyV] checked 1680000 candidates (len=4, last=89jx)
[DyV] checked 1685000 candidates (len=4, last=9det)
[DyV] checked 1690000 candidates (len=4, last=9g9p)
[DyV] checked 1695000 candidates (len=4, last=9k4l)
[DyV] checked 1700000 candidates (len=4, last=9ozh)
[DyV] checked 1705000 candidates (len=4, last=9sud)
[DyV] checked 1710000 candidates (len=4, last=9wo9)
[DyV] checked 1715000 candidates (len=4, last=90j5)
[DyV] checked 1720000 candidates (len=4, last=94el)
[DyV] checked 1725000 candidates (len=4, last=979x)
[RESULT] No encontrado (method=DyV) time=1.3769s checks=1727604
[STATS] Time total: 1.3863s | Algorithm time: 1.3769s | Checks: 1727604 | Time/check: 7.970167e-07s
[THEORY] Temporal: O(k^n) (divide la generación por mitades, no reduce la exponencialidad) | Espacial: O(n) pila recursiva
```

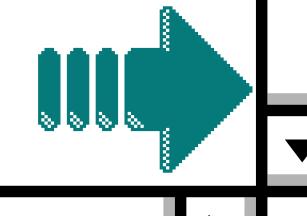
Tiempos

1.37.69 segundos

DyV

El enfoque de divide y vencerás, esta basado en la partición: el espacio de búsqueda puede dividirse en subespacios independientes (por ejemplo, por prefijos).

Start



Página 5 de 11

Vuelve a la página Agenda

Integración de las técnicas trabajadas.

Cracking Modes - Demo académico (v2)

Target (plain o hash): plain

Hash: sha256

Modo: Fuerza Bruta (FB) DyV (Divide & Vencerás) Prog Dinámica (edits desde wordlist) Voraz (Prim + MST)

Wordlist: C:/Users/PCREATIORS/Desktop/5TO SEMESTRE CUCEI ICOM/ANALISIS DE ALGORITMOS/EXPO CODIGO P/

Alphabet: abcdefghijklmnopqrstuvwxyz0123456789

Max len (FB/DyV): 4 PD: max edits: 2

Prim: k-neighbors: 6 Prim: top N start: 3

Candidate limit (global, 0 = no limit): 0 Random pwd len: 6

```
[ProgDin] checked 76000 candidates (seed=satanesmipastor last=satanesmipastor)
[ProgDin] checked 78000 candidates (seed=satanesmipastor last=batañesmiupastor)
[ProgDin] checked 80000 candidates (seed=satanesmipastor last=datanejsmipastor)
[ProgDin] checked 82000 candidates (seed=siguesleyendoesto? last=aicguesleyendoesto?)
[ProgDin] checked 84000 candidates (seed=siguesleyendoesto? last=cigujsleyendoesto?)
[ProgDin] checked 86000 candidates (seed=viejosabroso87 last=viejosabroso87b)
[ProgDin] checked 88000 candidates (seed=viejosabroso87 last=biejosabroso87s)
[ProgDin] checked 90000 candidates (seed=viejosabroso87 last=diejosabro087)
[ProgDin] checked 92000 candidates (seed=voyaprenderlefuegoacucei last=aoypare6derlefuegoacucei)
[ProgDin] checked 94000 candidates (seed=voyaprenderlefuegoacucei last=boyaprenderlefuepoacucei)
[ProgDin] checked 96000 candidates (seed=warhammer40k last=aaryammer40k)
[ProgDin] checked 98000 candidates (seed=warhammer40k last=pcarhammer40k)
[ProgDin] checked 100000 candidates (seed=warhammer40k last=earhammeru40k)
[ProgDin] checked 102000 candidates (seed=welcome last=celcomej)
[ProgDin] checked 104000 candidates (seed=welcome last=hzlcome)
[RESULT] No encontrado (method=ProgDin) time=4.9082s checks=105021
[STATS] Time total: 4.9111s | Algorithm time: 4.9082s | Checks: 105021 | Time/check: 4.673557e-05s
[THEORY] Temporal: Depende: O(m * growth(edits)) ~ O(m * k^e) | Espacial: O(k^e) por semilla (limitada con per_seed_limit)
```

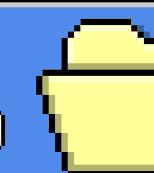
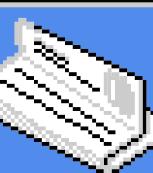
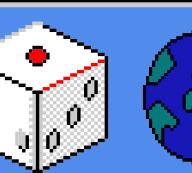
Tiempos

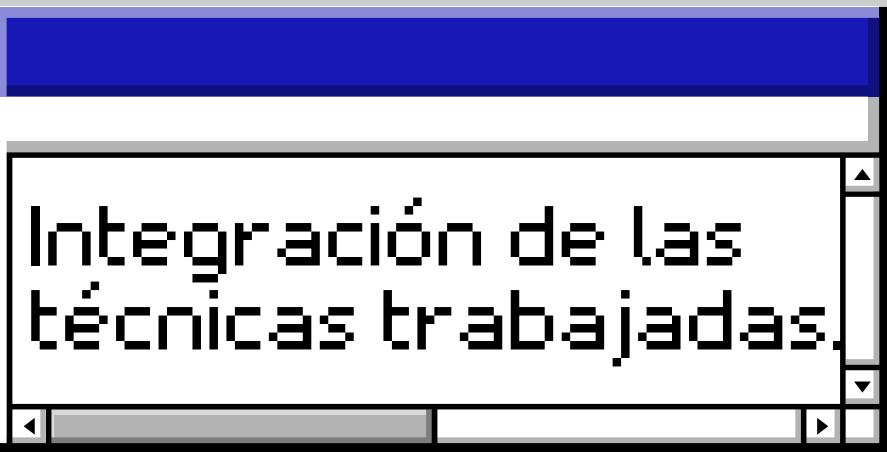
4.9082 segundos

Programación Dinámica

La programación dinámica se basa en comenzar con un conjunto inicial de contraseñas posibles y, desde ese punto, generar nuevas variantes mediante operaciones de edición como insertar, borrar o reemplazar caracteres. Por eso se considera un enfoque "dinámico", ya que modifica progresivamente las cadenas para acercarse a la solución.

Start





Cracking Modes - Demo académico (v2)

Target (plain o hash): **lemanruss** plain
 (si pones texto plano, selecciona 'plain' checkbox)

Hash: **sha256**

Modo: Fuerza Bruta (FB) DyV (Divide & Vencerás) Prog Dinámica (edits desde wordlist) Voraz (Prim + MST)

Wordlist: **C:/Users/PCREATHORS/Desktop/5TO SEMESTRE CUCEI ICOM/ANALISIS DE ALGORITMOS/EXPO CODIGO P**

Alphabet: **abcdefghijklmnopqrstuvwxyz0123456789**

Max len (FB/DyV): **4** PD: max edits: **2**
6 Prim: top N start: **3**
 Candidate limit (global, 0 = no limit): **0** Random pwd len: **6**

```
[ProgDin] checked 94000 candidates (seed=voyaprenderlefuegoacucei last=boyaprenderlefuepoacucei)
[ProgDin] checked 96000 candidates (seed=warhammer40k last=aaryammer40k)
[ProgDin] checked 98000 candidates (seed=warhammer40k last=pcarhammer40k)
[ProgDin] checked 100000 candidates (seed=warhammer40k last=earhammeru40k)
[ProgDin] checked 102000 candidates (seed=welcome last=celcomej)
[ProgDin] checked 104000 candidates (seed=welcome last=hzlcome)
[RESULT] No encontrado (method=ProgDin) time=4.9082s checks=105021
[STATS] Time total: 4.9111s | Algorithm time: 4.9082s | Checks: 105021 | Time/check: 4.673557e-05s
[THEORY] Temporal: Depende: O(m * growth(edits)) ~ O(m * k^e) | Espacial: O(k^e) por semilla (limitada con per_seed_limit)
[GUI] Target plain -> lemanruss (hash sha256: c283234b22244ec87ffec4cc930a3e6e22e5e36beef5908502894490db4cdd64)
[GUI] Starting mode PRIM...
[GUI] Running Voraz (Prim + MST)...
[RESULT] Found password: lemanruss (method=PrimMST) time=0.0103s checks=6
[STATS] Time total: 0.0123s | Algorithm time: 0.0103s | Checks: 6 | Time/check: 1.722250e-03s
[THEORY] Temporal: O(n^2) construcción (distancias) + O(n log n) Prim; recorrido O(n) | Espacial: O(n + e) para grafo
[GUI] Abriendo ventana de visualización MST...
```

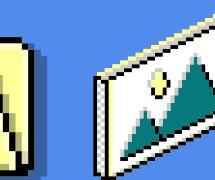
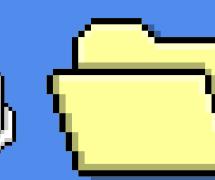
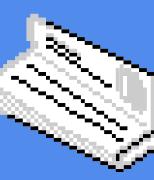
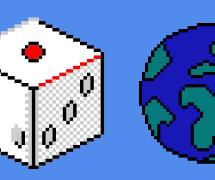
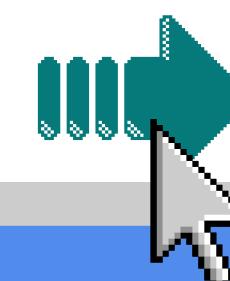
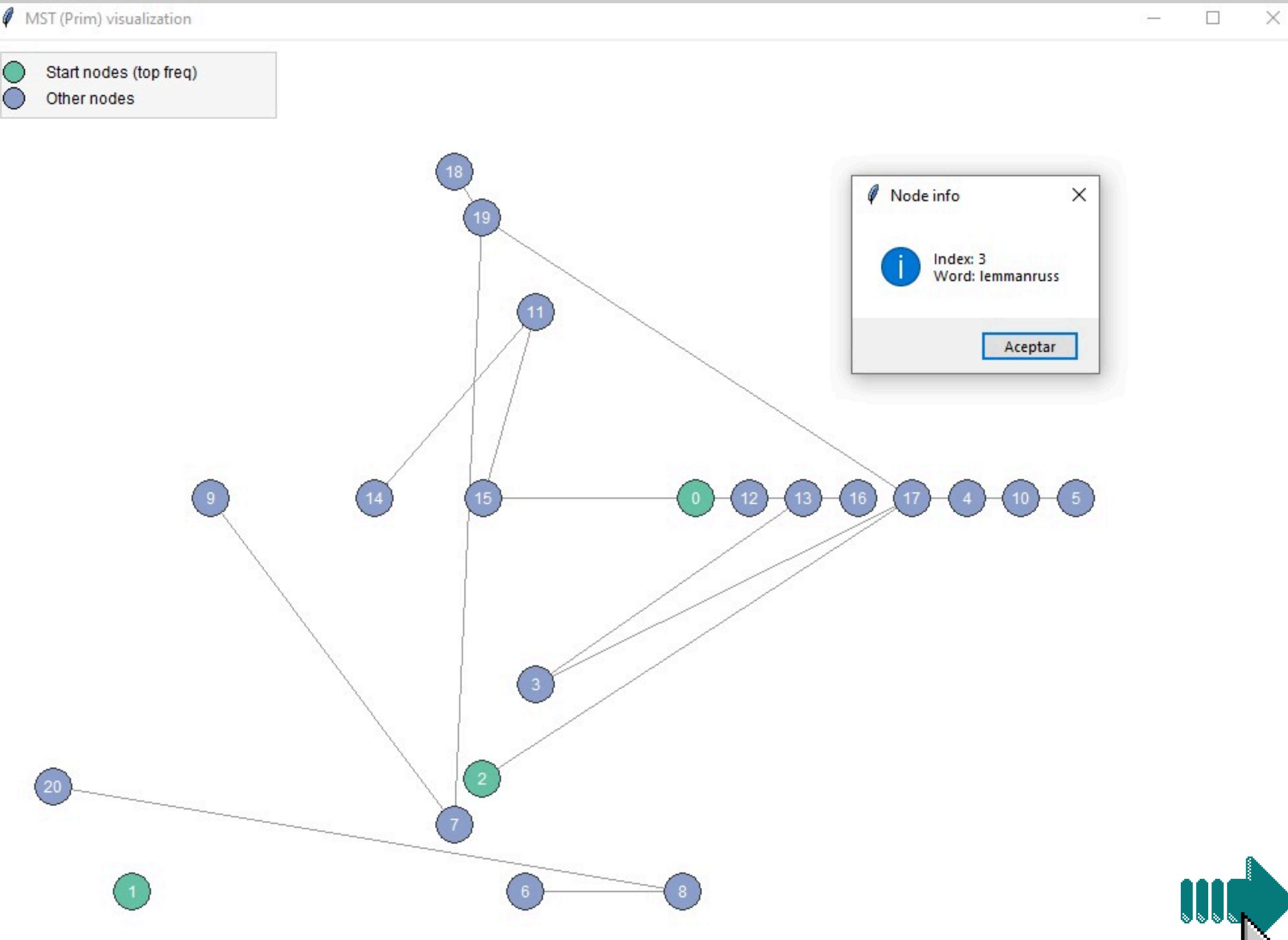
Tiempos

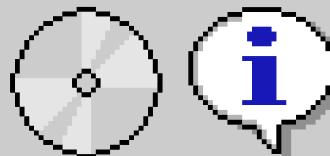
0.0103 segundos



PRIM con Voraz

Funciona trazando el MST o árbol de expansión menor dentro de un grafo construido tomando como base las posibles contraseñas del wordlist, recorriendo los candidatos que son representados por nodos.



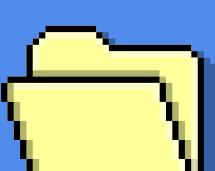
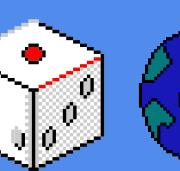


Resultados y comparativas



[Vuelve a la página Agenda](#)

	Fuerza Bruta	D y V	Programación Dinámica	PRIM con Voraz
Tiempos	1.3741 segundos	1.37.69 segundos	4.9082 segundos	0.0103 segundos

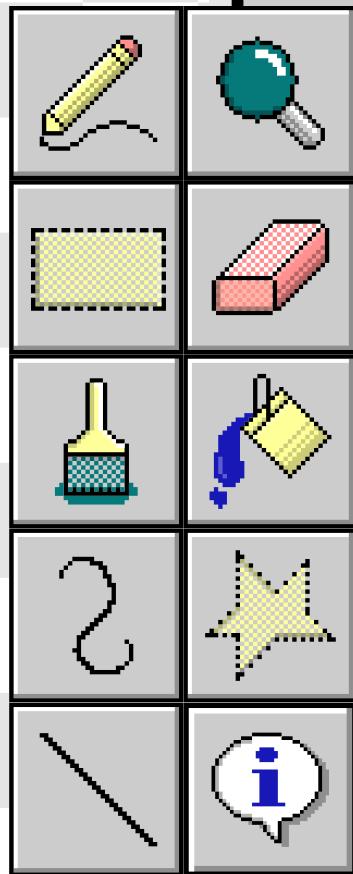


Conclusiones

Como puede notarse, si bien es verdad que, a nuestro algoritmo base, se le puede aplicar diversos enfoques o técnicas algorítmicas, cierto resulta también que, cada una de estas tiene sus ventajas y desventajas, traducidas en complejidad temporal y/o espacial.

Por lo anterior, consideramos que, el punto principal reside en, elegir el algoritmo que consuma menores recursos o bien, realice la función deseada con mayor eficiencia.

Consideramos que nuestro proyecto final puede ser catalogado como exitoso.



¡Muchas gracias!

Equipo: E-A-B-M-O-D-E-L