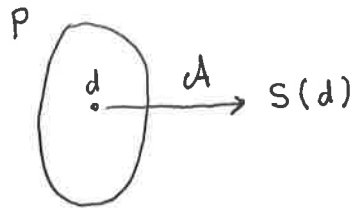
PRECONDICIONAMIENTO

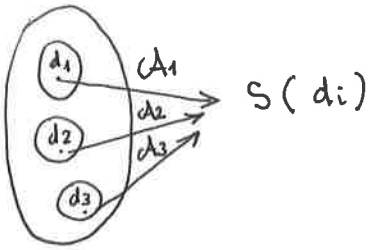
Procesamiento parcial por anticipado. Coste adelantado a amortizar.

Algoritmos parciales precocinados.

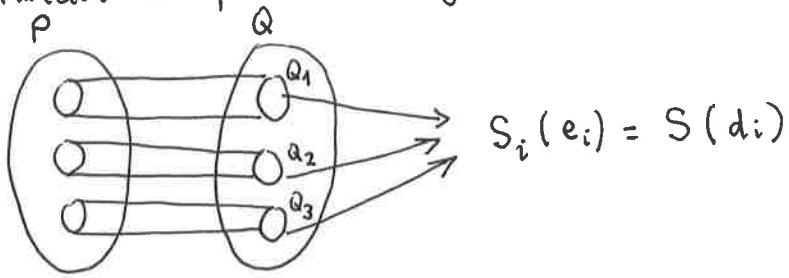
Situación de partida: Problema general, algoritmo genérico.



Distinción de casos y aplicación de algoritmos específicos.



Transformación de problemas (y datos) y resolución de los mismos



Ejemplo 1: Búsquedas en colecciones de datos

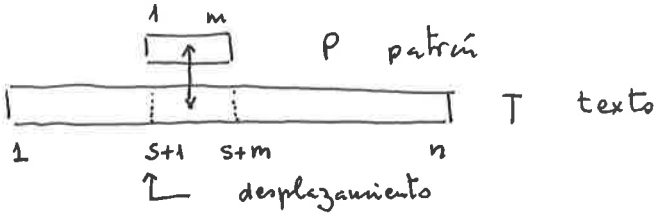
Preprocesamiento: ordenación de las colecciones sobre las que se repiten las búsquedas.

Ejemplo 2: Resolución de sistemas de ecuaciones $Ax = b$

Preprocesamiento: cálculo de A^{-1} si se han de resolver muchos sistemas correspondientes a una misma matriz.

BUSQUEDA de PATRONES en TEXTOS

- Encaje de patrón



- Prefijos de P : $P_k = P[1..k]$; $P_m = P$; $P_0 = \varepsilon$
- Buscamos los s | $P_m \sqsupseteq T_{s+m}$

Algoritmo ingénuo

Comparamos hasta terminar o fallar. Repetimos en la siguiente casilla.

Coste : Peor caso $m*(n-m)$; Mejor caso n.

Algoritmo de Rabin - Karp

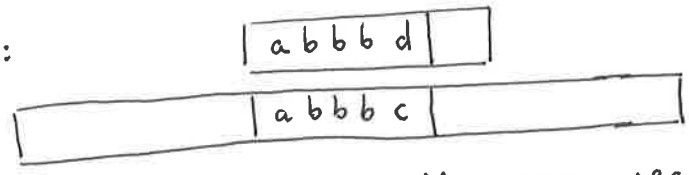
Se trata de reducir "mágicamente" la comparación de P y cada fragmento $T[s+1 \dots s+m]$ realizándola en un tiempo unidad. Por tiempo unidad entenderemos la comparación de dos "enteros - máquina".

- Dado un alfabeto $\mathcal{A}$ con d elementos podemos visualizar las palabras de A^m como números escritos en base d
- Como esos números pueden ser arbitrariamente grandes los "compactificamos" trabajando "mod q", donde q es un primo grande, pero que permita que todas las operaciones necesarias se puedan hacer sin rebasar la capacidad de los enteros de la máquina. Llamaremos "firmas" a los valores así obtenidos.
- Observamos que es muy fácil calcular la firma t_{s+1} de $T[s+1 \dots s+m]$ a partir de la de $T[s..s+m-1]$:
  Restamos $(d^{m-1} \cdot T[s]) \bmod q$ y sumamos $T[s+m] \bmod q$.
  Al efecto tendremos precalculado $d^{m-1} \bmod q$
- Inicialmente calculamos la firma de P y la de T_m y a partir de ahí recorremos las posibles $n-m+1$ posiciones iniciales de P en T.
- Cada fallo prueba un encaje inválido, y cada éxito "casi" prueba un encaje correcto (en muy pocos casos incorrectos) mediante una comparación de las cadenas originales.
- El tiempo esperado asumiendo pocos éxitos (una constante, c) y una distribución uniforme de las firmas es lineal en n.

Búsqueda de patrones guiada por un autómata

- La idea es que cada fallo tras haber comparado con éxito un prefijo P_q de P nos permite concluir que tampoco podemos encajar la secuencia en bastantes de las posiciones siguientes.

Ejemplo:



Como $a \neq b$, tras haber fallado el encaje con $c \neq d$ podemos avanzar con seguridad hasta esa casilla para seguir probando.

- En general, dado $q < m$ y una palabra de la forma $P_q \circ \langle x \rangle$ definimos $\sigma(P_q \circ \langle x \rangle) = \max\{k \mid P_k \sqsupseteq P_q \circ \langle x \rangle\}$
- El preprocesamiento consiste en precalcular la tabla de σ, que en principio tiene $m \cdot |\Sigma|$ entradas, siendo Σ el alfabeto sobre el que se trabaja, si bien podríamos reducirnos a $\Sigma_P = \{P[i] \mid i \in 1..m\}$ pues alcanzado cualquier $x \notin \Sigma_P$ tendríamos trivialmente $\sigma(P_q \circ \langle x \rangle) = 0$.
- El procesamiento del texto consiste en recorrerlo aplicando la tabla de σ y cada vez que alcancemos el estado m tendríamos un ajuste de P.
- El cálculo de la tabla "a lo bruto" requeriría tiempo $O(m^3 |\Sigma_P|)$, pero realizándolo con astucia podemos rebajarlo a $O(m |\Sigma_P|)$.

Algoritmo KMP (Knuth, Morris y Pratt)

- En esencia es una "simplificación" del algoritmo anterior a costa de "ignorar" la información que pueda darnos el valor leído x, que cause un "fallo local" que no permita incrementar el estado en el que nos encontramos.
- Dado P de longitud m, definimos $\Pi : 1..m \to 0..m+1$ mediante

$$\Pi[q] = \max\{k < q \mid P_k \sqsupset P_q\}$$

Longitud del mayor prefijo de P que es sufijo propio de P_q.