Programación Dinámica

Carlos Linares López

Grupo de Planificación y Aprendizaje (PLG)
Departamento de Informática
Universidad Carlos III de Madrid

18 de septiembre de 2017

Definición

Intuición

La *Programación Dinámica* es una técnica *bottom-up* que sugiere almacenar resultados intermedios que pueden reusarse para calcular el valor óptimo global de un problema

- Es especialmente útil cuando puede establecerse un orden en la solución de subproblemas

Procedimiento general

1. Descomponer el problema de optimización global en subproblemas similares y caracterizar su estructura
2. Definir recursivamente la solución óptima del subproblema
3. Calcular los valores de las soluciones óptimas de los subproblemas de abajo hacia arriba (*bottom-up*)
4. Calcular la solución óptima del problema global a partir de los resultados intermedios

En algunos casos puede ser preciso habilitar mecanismos adicionales para recuperar la solución

Serie de Fibonacci I

Problema

Calcula el término enésimo de la serie de Fibonacci, $F(n)$ definida por la relación:

$$F(n) = F(n-1) + F(n-2)$$

cuyos casos base son:

$$F(0)=0$$
$$F(1)=1$$

Serie de Fibonacci II

Una implementación ingenua resolvería el problema codificando directamente la relación anterior:

```
long fibonacci_0 (int n)
{
  if (n==0) return 0;
  if (n==1) return 1;
  return fibonacci_0 (n-1) + fibonacci_0 (n-2);
}
```

Serie de Fibonacci III

Sin embargo, eso conduce a una cantidad exponencial de evaluaciones: ϕ^n

$$
\begin{tikzcd}
 & & & F(4) \arrow[dl] \arrow[dr] & & \\
 & & F(3) \arrow[dl] \arrow[dr] & & F(2) \arrow[dl] \arrow[dr] & \\
 & F(2) \arrow[dl] \arrow[dr] & & F(1) & & F(0) \\
F(1) & & F(0) & & &
\end{tikzcd}
$$

$F(0)$, $F(1)$ y $F(2)$ se evaluan dos veces cada uno y el caso se agrava para valores mayores de n

Serie de Fibonacci IV

Sin embargo, almacenando los valores intermedios:

```
long fibonacci_aux (int n)
{
   if (f[n]==UNKNOWN)
      f[n]=fibonacci_aux (n-1) + fibonacci_aux (n-2);
   return f[n];
}
long fibonacci_1 (int n)
{
   int i;
   f[0]=0;
   f[1]=1;
   for (i=2; i≤n; f[i++]=UNKNOWN);
   return fibonacci_aux (n);
}
```

Serie de Fibonacci V

Con Programación Dinámica ni siquiera es preciso en este caso usar una tabla y se eliminan todas las llamadas recursivas

```
long fibonacci_2 (int n)
{
   int i,z;
   int x = 0;
   int y = 1;
   if (n<2)
      return n;
   for (i=2;i≤n;i++) {
      z=x+y;
      x=y;
      y=z;
   }
   return z;
}
```

All-pairs shortest-path

Problema

Dado un grafo $G(V, E)$ calcula el coste del camino más corto entre todos los pares de vértices i y j, $D(i, j)$

Algoritmo Floyd-Warshall I

Dada la secuencia de matrices, $D^0, D^1, D^2, \ldots$ donde D^0 es la matriz de adyacencia y D^k contiene la distancia del camino más corto entre cualquier par de nodos usando únicamente vértices en el conjunto $\{1, 2, \ldots, k\}$, $D^k_{i,j}$ se puede definir como:

$$D^k_{i,j} = \text{mín}\left\{ \underbrace{D^{(k-1)}_{i,j}}_{\text{No usa } k}, \underbrace{D^{(k-1)}_{i,k} + D^{(k-1)}_{k,j}}_{i \to k \text{ y } k \to j} \right\}$$

El cálculo de la distancia mínima entre todos los pares concluye después de N productos (con $|V| = N$) de la matriz D de dimensión N^2, con una complejidad de $\Theta(N^3)$

Algoritmo Floyd-Warshall II

Para reconstruir la solución, basta con almacenar para cada vértice el índice intermedio más alto que debe atravesarse para llegar a otro vértice. La tabla resultante contiene $|V|^2$ vertices for all pairs $(i, j) \in V \times V$

Conclusiones

- Las expresiones de recurrencia deben forzar la simplicidad de la solución general
- El tiempo de ejecución depende de:
  - El número de soluciones parciales que deben almacenarse
  - El tiempo que tarda en resolverse cada problema parcial

Bibliography

- Steven S. Skiena
  The Algorithm Design Manual
  Capítulo 8
  Springer, 2008, Second Edition

- Dejan Zivkovic
  Foundations of Algorithm Analysis and Design
  Capítulo 7
  Verlag Dr. Müller, 2009

- Stefan Edelkamp and Stefan Schroedl
  Heuristic Search: Theory and Applications
  Sección 3.1.7
  Morgan Kaufmann, 2010