Metaheurísticas

Seminario 4. Técnicas basadas en trayectorias para el Problema de Asignación Cuadrática (QAP) y el Aprendizaje de Pesos en Carácterísticas (APC). Evolución Diferencial para el APC

1. Trayectorias Simples
   - Esquema General del Algoritmo de Enfriamiento Simulado
   - Un Algoritmo de Enfriamiento Simulado para el QAP y el APC
2. Trayectorias Múltiples
   - Esquema General de los Algoritmos GRASP e ILS
   - Un Algoritmo GRASP para el QAP
   - Un Algoritmo ILS para el QAP y el APC
3. Evolución Diferencial
   - Esquema General de un Algoritmo de Evolución Diferencial
   - Algoritmo de Evolución Diferencial para el APC

Algoritmo de Enfriamiento Simulado

Procedimiento Simulated Annealing *(Δf para minimizar)*

Start

$T \leftarrow T_0$; $s \leftarrow$ GENERATE(); Best Solution $\leftarrow s$;

Repeat

→ Cuantos vecinos para una temperatura ó nº fijo o nº de exitos

For *cont* = 1 to *L(T)* **do** /* Inner loop

Start

→ Genera un vecino aleatorio

$s' \leftarrow$ NEIGHBORHOOD_OP(s); /* A single move

$\Delta f = f(s') - f(s)$;

→ Si es mejor o el criterio aleatorio de metropolis se cumple lo cojo

If (($\Delta f < 0$) or ($U(0,1) \leq \exp(-\Delta f / k \cdot T)$)) then

$s \leftarrow s'$;

If COST(s) **is better than** COST(Best Solution)

then Best Solution $\leftarrow s$;

End

$T \leftarrow g(T)$; /* Cooling scheme. The classical one is geometric: $T \leftarrow \alpha \cdot T$

until ($T \leq T_f$); /* Outer loop

Temperatura mínima

Return(Best Solution);

End

Enfriamiento Simulado para el QAP

- **Representación**: Problema de asignación: una permutación $\pi=[\pi(1), \ldots, \pi(n)]$ en el que las posiciones del vector $i=1,\ldots,n$ representan las unidades y los valores $\pi(1), \ldots, \pi(n)$ contenidos en ellas las localizaciones. Permite verificar las restricciones

- **Operador de vecino de intercambio y su entorno**: El entorno de una solución π está formado por las soluciones accesibles desde ella a través de un movimiento de intercambio

  Dada una solución (asignación de unidades a localizaciones) se escogen dos unidades distintas y se intercambia la localización asignada a cada una de ellas ($Int(\pi,i,j)$):

$$\pi=[\pi(1), \ldots, \pi(i), \ldots, \pi(j), \ldots, \pi(n)]$$

$$\pi'=[\pi(1), \ldots, \pi(j), \ldots, \pi(i), \ldots, \pi(n)]$$

Enfriamiento Simulado para el QAP

- Exploración del vecindario: En cada iteración del bucle interno se genera una única solución vecina, **de forma aleatoria**, y se compara con la actual. **No se usa el don't look bits de la búsqueda local**

- Esquema de enfriamiento: esquema de Cauchy modificado

- Condición de enfriamiento *L(T)*: cuando se genere un número máximo de soluciones vecinas, *máx_vecinos*, o cuando se acepte un número máximo de los vecinos generados, *máx_éxitos*

- Condición de parada: cuando se alcance un número máximo de iteraciones o cuando el número de éxitos en el enfriamiento actual sea 0

Enfriamiento Simulado para el APC

- Representación: vector de números reales con pesos de características y la posibilidad de eliminarlas, igual que en prácticas anteriores
- Operador de generación de vecinos: mutación normal, como en la BL
- Exploración del vecindario: En cada iteración del bucle interno se genera una única solución vecina, de forma aleatoria, y se compara con la actual
- Esquema de enfriamiento: esquema de Cauchy modificado
- Condición de enfriamiento *L(T)*: cuando se genere un número máximo de soluciones vecinas, *máx_vecinos*, o cuando se acepte un número máximo de los vecinos generados, *máx_éxitos*
- Condición de parada: cuando se alcance un número máximo de iteraciones o cuando el número de éxitos en el enfriamiento actual sea 0

Procedimiento GRASP

Procedimiento GRASP

Repetir Mientras (no se satisfaga el criterio de parada)

S ← Construcción Solución Greedy Aleatorizada ()

S' ← Búsqueda Local (S)

Actualizar (S', *Mejor_Solución*)

Devolver (*Mejor_Solución*)

FIN-GRASP

Procedimiento GRASP

Construcción Solución Greedy Aleatorizada ()

- ✓ En cada iteración de su proceso constructivo de la solución, un algoritmo greedy básico:
  - ✓ construye una lista con los candidatos factibles (las posibles componentes a escoger de acuerdo con la solución construida hasta el momento y las restricciones del problema): **Lista de Candidatos (LC)**,
  - ✓ los evalúa de acuerdo a una función de selección (que mide su calidad/preferencia para ser escogidos), y
  - ✓ selecciona siempre el candidato de **mejor calidad** de la LC

- ✓ Los algoritmos GRASP añaden **aleatoriedad** al procedimiento anterior. La única diferencia es que en cada iteración:
  - ✓ No se consideran todos los candidatos posibles sino sólo los de mejor calidad: **Lista Restringida de Candidatos (LRC)**. El tamaño de esa lista puede ser fijo o variable en función de un umbral de calidad
  - ✓ El elemento seleccionado se escoge **aleatoriamente** de la RCL para inducir diversidad, independientemente de la calidad de los candidatos

Problema de Asignación Cuadrática (QAP)

- **Problema de la asignación cuadrática, *QAP***:

  Dadas n unidades y n localizaciones posibles, el problema consiste en determinar la asignación óptima de las unidades en las localizaciones conociendo el flujo existente entre las primeras y la distancia entre las segundas

$$QAP = \min_{S \in \Pi_N} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} \cdot d_{S(i)S(j)} \right)$$

donde:

- ✓ S es una solución candidata (una posible asignación de unidades a localizaciones) representada por una permutación de n elementos
- ✓ $f_{ij} \cdot d_{S(i)S(j)}$ es el coste de la asignación de la unidad u_i a la localización $S(i)$ y u_j a $S(j)$, calculado como el coste del recorrido del flujo que circula entre esas dos unidades i y j cuando están situadas en las localizaciones S(i) y S(j)

Procedimiento GRASP para el QAP

Variante de: Y. Li, P.M. Pardalos, M.G.C. Resende. A greedy randomized adaptive search procedure for the quadratic assignment problem. In: P.M. Pardalos and H. Wolkowicz (Eds.), Quadratic assignment and related problems, 1994, pp. 237-261

- Los candidatos son las asignaciones unidad-localización factibles. La función de selección mide su coste de interacción $f_{ij} \cdot d_{kl}$ (a menor coste, más preferencia). Se busca asociar unidades de gran flujo con localizaciones céntricas

- El procedimiento de construcción de la solución greedy aleatorizada tiene dos etapas:

  - Etapa 1: Se toma la primera decisión, en la que **se asignan conjuntamente dos unidades a dos localizaciones en un solo paso**. Se consideran los potenciales de flujo y distancia de unidades y localizaciones para escoger el par de asignaciones en una decisión *greedy* aleatorizada

  - Etapa 2: Se toman las n-2 decisiones restantes **asignando una unidad a una localización en cada paso**

    Se escoge la asignación factible con el mínimo coste de interacción con respecto a las asignaciones realizadas hasta ahora de acuerdo a una decisión *greedy* aleatorizada

Procedimiento GRASP para el QAP

Etapa 1:

- Se dispone de dos LCs, una para las unidades LC_u y otra para las localizaciones LC_l. Ambas listas contienen inicialmente las n unidades y las n localizaciones, respectivamente

- La función de selección será el potencial de flujo/distancia de las unidades, calculado como:

$$\hat{f}_i = \sum_{j=1}^{n} (f_{ij} + f_{ji}) \quad ; \quad \hat{d}_k = \sum_{l=1}^{n} (d_{kl} + d_{lk})$$

- En el *greedy*, la decisión sería escoger el mejor candidato de cada LC, es decir, asignar la unidad de mayor potencial de flujo con la localización de menor distancia potencial

Procedimiento GRASP para el QAP

- En el GRASP, las dos LRCs, LRC_u y LRC_l, son de tamaño variable e incluyen todos los candidatos de la LC correspondiente cuyo coste iguala o supera un umbral de calidad:
  - LRC_u: Como los candidatos son mejores cuánto más alto sea el potencial de flujo, el umbral es μ = **d_max -** $\alpha \cdot$ (d_max – d_min), donde d_max es el mayor valor de flujo potencial de los candidatos de LC_u y d_min el menor
  - LRC_l: Como los candidatos son mejores cuánto más bajo sea el potencial de distancia, el umbral es μ = **d_min +** $\alpha \cdot$ (d_max – d_min), donde d_max es la mayor distancia potencial de los candidatos de LC_l y d_min la menor
- Se escogen aleatoriamente dos candidatos de cada LRC, se asocian el primero escogido de la LRC_u u_i con el primero de la LRC_l l_k; y luego el segundo de la LRC_u u_j con el segundo de la LRC_l l_l
- Se guardan las dos asignaciones unidad-localización generadas de este modo en la solución parcial, $S = \{(u_i,l_k), (u_j,l_l)\}$

1er Paso

$u_1, u_2, \dots, u_n$

$\hat{g}_1, \hat{g}_2, \dots, \hat{g}_n$

↳ Mejor = Mayor valor

$l_1, l_2, \dots, l_n$

$\hat{d}_1, \hat{d}_2, \dots, \hat{d}_n$

↳ Mejor = menor valor

→ LCV = { $u_1, u_2, \dots, u_n$ } 370, 320, 295, 200 ...

↳ Ordenar

↳ Q. diferencias calidad

↳ Coger en el humbral q. dif = 70

370 → Casa mas cara

− 70 → Diferencia

300

2.º Paso:

$u_1 \Rightarrow l_2$

$u_{13} \Leftrightarrow l_{12}$

n-2 decisiones:

No las cogemos pq ya hemos cogido u_1

LC = { (1,1), (1,2), ..., (1,n),

Para u_2 → (2,1), (2,2), ..., (2,n),

:

(n,1), (n,2), ..., (n,n) }

→ Como es un grasp, cogemos las mejores y escogemos aleatoria mente.

Procedimiento GRASP para el QAP

Etapa 2:

- En la segunda etapa, se escoge una sola asignación unidad-localización en cada paso
- La LC está formada por todas las asignaciones factibles (u_i,l_k), es decir, todos los pares unidad-localización en los que **ninguna de ellas** esté incluida aún en la solución parcial S = {(u_{i1},l_{k1}), …, (u_{iz},l_{kz})}
- La segunda etapa comienza con |S|=2, las dos asignaciones realizadas en la primera etapa
- La función de selección *greedy* mide el coste de añadir la asignación candidata (u_i,l_k) a las asignaciones ya realizadas, almacenadas en la solución actual S, calculado como:

$$C_{ik} = \sum_{(j,l)\in S} f_{ij} \cdot d_{kl}$$

Procedimiento GRASP para el QAP

- Los candidatos con menor valor de la función son preferibles
- De nuevo, la LRC es de tamaño variable e incluye todas las asignaciones factibles de LC cuyo coste es menor o igual que el umbral de calidad μ = Cmin + $\alpha \cdot$ (Cmax - Cmin), donde Cmin es el menor coste de los candidatos de LC y Cmax el mayor
- Se escoge aleatoriamente una asignación unidad-localización candidata de la LRC y se añade a la solución parcial $S \leftarrow S \cup \{(u_i,l_k)\}$
- En cada nuevo paso del algoritmo hay que actualizar la LC, eliminando las asignaciones consideradas en el paso anterior, y construir la nueva LRC recalculando los costes para los candidatos factibles restantes y aplicando el umbral para filtrar los candidatos a emplear
- El proceso constructivo termina tras n-2 pasos de la segunda etapa, cuando ya se han tomado las n decisiones necesarias

Procedimiento ILS

Comienzo-ILS

S_o ← Generar-Solución-Inicial

S ← Búsqueda Local (S_o)

Repetir

S' ← Modificar (S, historia) **%Mutación**

S'' ← Búsqueda Local (S')

S ← Criterio-Aceptación (S, S'', historia)

Actualizar (S, *Mejor_Solución*)

Hasta (Condiciones de terminación)

Devolver *Mejor_Solución*

Fin-ILS

ILS para el QAP

- Representación de orden: permutación $\pi=[\pi(1), \ldots, \pi(n)]$ en el que las posiciones del vector $i=1,\ldots,n$ representan las unidades y los valores $\pi(1), \ldots, \pi(n)$ contenidos en ellas las localizaciones

- Solución inicial: aleatoria

- Operador de mutación: Usaremos el operador de vecino de sublista aleatoria de tamaño fijo t, consistente en seleccionar una cadena consecutiva de asignaciones y reasignarlas aleatoriamente

  Para provocar un cambio brusco, obligaremos a que la sublista tenga un tamaño importante: $t = n/4$

- Algoritmo de búsqueda local: la BL-QAP de la Práctica 1

- Criterio de aceptación: se sigue el "criterio del mejor", siempre se aplica la mutación sobre la mejor solución encontrada hasta ahora

ILS para el APC

- Representación real: Vector real W de tamaño *n,* donde $w_i \in [0, 1]$, con la posibilidad de eliminar características
- Solución inicial: aleatoria
- Operador de mutación: Cada vez que se muta, se aplica la mutación normal a $t=0.1 \cdot n$ características con mayor σ
- Función objetivo: La agregación de las tasas de clasificación y de reducción empleada hasta el momento
- Algoritmo de búsqueda local: la BL-APC de la Práctica 1
- Criterio de aceptación: se sigue el "criterio del mejor", es decir, siempre se aplica la mutación sobre la mejor solución encontrada hasta el momento

Evolución Diferencial

Procedure DE/rand/1 – Rec. Binomial {
t = 0;
Initialize Pop(t); /* of |Pop(t)| Individuals */
Evaluate Pop(t);
While (Not Done)
{for i = 1 to |Pop(t)| do
{parent1, parent2, parent3} = Select_3_Parents(Pop(t));
for k = 1 to n do /* n genes per Individual */
if (random < CR) /* CR is crossover constant in [0,1] */
$Offspring_{ik} = parent1_{ik} + F\cdot(parent2_{ik} - parent3_{ik})$;
else
$Offspring_{ik} = Individual_{ik}$ in Pop(t);
end /* for k */
Evaluate($Offspring_i$);
end /* for i */
Pop(t+1) = {j | $Offspring_j$ is_better_than $Individual_j$} ∪
{k | $Individual_k$ is_better_than $Offspring_k$};
t = t + 1; }

Evolución Diferencial para APC

- Población inicial: generada aleatoriamente.

- Modelos a implementar:

$$\text{DE/rand/1:} \quad \mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \cdot \left(\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}\right)$$

$$\text{DE/current-to-best/1:} \mathbf{V}_{i,G} = \mathbf{X}_{i,G} + F \cdot \left(\mathbf{X}_{best,G} - \mathbf{X}_{i,G}\right) + F \cdot \left(\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}\right)$$

- Recombinación Binomial:

$$u_{j,i,g} = \begin{cases} v_{j,i,g} & \text{if } \text{rand}_j[0,1] \leqslant Cr \text{ or } j = j_{\text{rand}} \\ x_{j,i,g} & \text{otherwise} \end{cases}$$

- Reemplazamiento: uno a uno

- Función objetivo: La agregación de las tasas de clasificación y de reducción empleada hasta el momento