UNIVERSIDAD TORCUATO DI TELLA

Tecnología Digital V: Diseño de algoritmos
Licenciatura en Tecnología Digital
Primer Semestre, 2024

TP3 - Heurísticas para el Problema del Viajante de Comercio Asimétrico (ATSP)

El contexto

El Problema del Viajante de Comercio Asimétrico (ATSP, por sus siglas en inglés) es una variación del clásico Problema del Viajante de Comercio (TSP). En el ATSP, un vendedor debe visitar un conjunto de ciudades exactamente una vez y regresar a la ciudad de origen, minimizando la distancia total recorrida. La característica distintiva del ATSP es que las distancias entre las ciudades no son simétricas; es decir, la distancia de la ciudad A a la ciudad B puede ser diferente de la distancia de la ciudad B a la ciudad A. A diferencia del TSP, el ATSP captura naturalmente situaciones donde los costos de los arcos son asimétricos, como por ejemplo al movernos en ciudades donde las calles no son todas de doble circulación, y por lo tanto el camino de entre dos puntos A y B puede diferir del de B a A.
En la práctica, el ATSP tiene aplicaciones en diversas áreas. Una de las áreas clásicas de aplicación es *Logística y Transporte*, donde el ATSP y sus variantes tienen una traducción directa en problemas de optimización de flota para la reducción de costos. Sin embargo, también tiene aplicaciones en otras áreas tales como la perforación en circuitos integrados, la optimización de códigos genéticos, problemas de recolección en depósitos, entre otras.
El TSP y el ATSP pertenecen a la clase $\mathcal{NP}$-hard, y han sido estudiados en gran detalle por la comunidad científica durante la últimas décadas. A pesar de su dificultad, actualmente se dispone de algoritmos extremadamente eficientes que permiten resolver a optimalidad instancias de miles de clientes en unos pocos minutos (ver, por ejemplo, Concorde). Los desarrollos propuestos para el TSP y el ATSP han sido luego generalizados a otros problemas, permitiendo avanzar el conocimiento no solo para el problema en particular sino para la Optimización Combinatoria en general.

El modelo

Sea $D = (V, A)$ un grafo dirigido completo, con $V = \{1, \ldots, n\}$ el conjunto de vértices y $A = \{(i, j) : i, j \in V, i \neq j\}$ y el conjunto de aristas. Sea $c : A \longrightarrow \mathbb{R}_{\geq 0}$ la función de costos de los arcos. Sea $T = (v_1, v_2, \ldots, v_n)$ un circuito en D, con $v_i \in V$, definimos el costo de T, $c(T)$, como

$$c(T) = \sum_{i=1}^{n-1} c(v_i, v_{i+1}) + c(v_n, v_1).$$

El ATSP consiste en encontrar un *Circuito Hamiltoneano* (es decir, un circuito que vista cada vértice en V exactamente una vez) de costo mínimo.

Los datos

Contamos con instancias de benchmark para el ATSP de la reconocida TSPLIB, ampliamente utilizada en investigaciones relacionadas al TSP y sus variantes. Los archivos son de texto plano, siguiendo el formato estándar de la librería (documentación).

El trabajo

El trabajo práctico consiste en implementar disintas heurísticas para el ATSP, y evaluarlas utilizando las instancias de benchmark de TSPLIB. Los grupos pueden ser de máximo 2 personas. El trabajo práctico puede ser implementado en Python o C++, a decidir por el grupo. Se pide:

1. **Heurísticas constructivas.** Implementar (al menos) dos heurísticas constructivas distintas para el ATSP. Analizar la complejidad de los algoritmos, que debe ser consistente con la implementación.

2. **Operadores de búsqueda local.** Implementar (al menos) dos operadores de búsqueda local para el ATSP. Analizar la complejidad de los algoritmos, que debe ser consistente con la implementación.

3. **Métodos.** Proponer un método que combine una heurística constructiva con los operadores de búsqueda local.

4. **Experimentación y discusión.** Realizar una experimentación exhaustiva con las instancias de benchmark provistas, comparando los métodos propuestos. Los métodos a compraar deben ser (al menos) los siguientes:

   - Cada heurística constructiva de manera independiente.
   - Heurística constructiva + operador de búsqueda local. Ejemplo; vecino más cercano + relocation; vecino más cercano + swap.
   - Heurística constructiva (alguna) + combinación de operadores de búsqueda local.

   La experimentación debe considerar la calidad de las soluciones[1] así como también el tiempo de ejecución. Respecto al método propuesto en el ítem anterior, el diseño del mismo debe estar justificado por la experimentación.

5. **Informe, presentación de resultados y delivery del código.** La descripción de los algoritmos, los operadores, las decisiones de diseño, la implementación, el testing realizado, la presentación de resultados, instrucciones de compilación y ejecución.

Modalidad de entrega

Se pide presentar el modelo y la experimentación en un informe de máximo 8 páginas que contenga:

- introducción al problema,
- descripción de los algoritmos implementados, según corresponda,
- consideraciones generales respecto a la implementación, incluyendo dificultades que hayan encontrado,
- resumen de resultados obtenidos en la experimentación,

[1] En caso de necesitarlo, en el siguiente link se encuentran los valores las mejores soluciones conocidas para cada instancia.

- conclusiones, posibles mejoras y observaciones adicionales que consideren pertinentes.

Junto con el informe debe entregarse el código con la implmenetación del modelo. El mismo debe ser entendible, incluyendo comentarios que faciliten su corrección y ejecución.

Fechas de entrega

Formato Electrónico: **Lunes 8 de julio de 2024, 23:59 hs**, enviando el trabajo (informe + código) vía el campus virtual.
Importante: El horario es estricto. Las entregas recibidas después de la hora indicada serán considerados re-entrega.