Informe de la Tercera Evaluación

Arnovi Antonio Jimenez Velasquez
Jerik David Hincapie Bedoya
Universidad Tecnológica de Pereira
IS845: Computación Blanda
Luis Fernando Galindres
Noviembre 12 del 2024

TABLA DE CONTENIDO

1. Introducción

En este informe, se analiza el rendimiento de un algoritmo de Optimización de Colonia de Hormigas (ACO) aplicado al Problema del Viajero (TSP). Para evaluar su desempeño, se han utilizado instancias seleccionadas de la biblioteca TSPLIB, comparando los resultados obtenidos con los valores óptimos conocidos.

Se han implementado varias combinaciones de parámetros del algoritmo, evaluando el impacto de estos sobre la calidad de las soluciones y el tiempo de ejecución.

2. Conceptos Utilizados

2.1 Problema del Viajero (TSP)

El TSP es un problema NP-difícil donde se busca encontrar la ruta más corta que visite un conjunto de ciudades exactamente una vez, regresando a la ciudad de origen. Este problema tiene importantes aplicaciones en logística y planificación de rutas.

2.2 Optimización de Colonia de Hormigas (ACO)

El ACO es un algoritmo basado en el comportamiento de las hormigas reales, las cuales encuentran caminos óptimos entre su colonia y las fuentes de alimento depositando feromonas. En el algoritmo, las hormigas construyen rutas, y las rutas exitosas (de menor distancia) incrementan la cantidad de feromonas, lo que aumenta la probabilidad de que futuras hormigas sigan esas rutas.

2.3 Parámetros del ACO

Número de hormigas (n_ants): Cantidad de hormigas que exploran soluciones.
Alpha (α): Influencia de la cantidad de feromona.
Beta (β): Influencia de la visibilidad (inverso de la distancia).
Rho (ρ): Tasa de evaporación de feromona.
Número de iteraciones (n_iterations): Máximo de iteraciones del algoritmo.

3. Resultados

A continuación, se presentan los resultados obtenidos para las instancias seleccionadas de TSPLIB. Se muestran los valores óptimos, las mejores soluciones encontradas por el ACO, el GAP (diferencia porcentual entre la solución obtenida y la mejor conocida), y el tiempo de ejecución.

3.1 Mejores Soluciones por Instancia

Instancia	Categoría	Distancia	GAP (%)	Tiempo (s)	n_ants	Alpha	Beta	Rho	Iteraciones	Estabilización
burma14	Pequeñas	3323.0	0.00	0.5678	20	1.5	0.5	0.5	2000	22
ulysses16	Pequeñas	6859.0	0.00	1.3384	20	1.5	1.0	0.1	1000	91
gr17	Pequeñas	2085.0	0.00	0.5606	10	1.0	1.5	0.3	1000	72
eil51	Medianas	438.0	2.82	10.9592	30	1.0	1.5	0.3	1000	97
berlin52	Medianas	7755.0	2.82	4.0340	20	1.5	1.5	0.3	1000	37
brazil58	Medianas	26136.0	2.92	8.8890	20	1.0	1.5	0.5	2000	127
eil101	Grandes	667.0	6.04	39.1897	30	1.5	1.0	0.1	2000	140
lin105	Grandes	15352.0	6.77	17.3868	20	1.0	1.5	0.5	1000	78
pr107	Grandes	47504.0	7.23	52.1697	30	1.5	1.5	0.1	2000	196

3.1.1 Captura resultados de ejecución

```
Mejores soluciones por instancia:
Instancia  Categoría  Distancia  GAP(%)  Tiempo(s)  n_ants alpha  beta  rho  iter  conv
-------------------------------------------------------------------------------------
burma14    pequeñas   3323.0     0.00    0.5678     20     1.5    0.5   0.5  2000  22
ulysses16  pequeñas   6859.0     0.00    1.3384     20     1.5    1.0   0.1  1000  91
gr17       pequeñas   2085.0     0.00    0.5606     10     1.0    1.5   0.3  1000  72
eil51      medianas   438.0      2.82    10.9592    30     1.0    1.5   0.3  1000  97
berlin52   medianas   7755.0     2.82    4.0340     20     1.5    1.5   0.3  1000  37
brazil58   medianas   26136.0    2.92    8.8890     20     1.0    1.5   0.5  2000  127
eil101     grandes    667.0      6.04    39.1897    30     1.5    1.0   0.1  2000  140
lin105     grandes    15352.0    6.77    17.3868    20     1.0    1.5   0.5  1000  78
pr107      grandes    47504.0    7.23    52.1697    30     1.5    1.5   0.1  2000  196
```

Notas:

- Para cada instancia, se presenta el mejor camino encontrado y los parámetros con los que se obtuvo.
- El GAP indica qué tan cerca estuvo el algoritmo de la solución óptima conocida.

3.2 Análisis Global

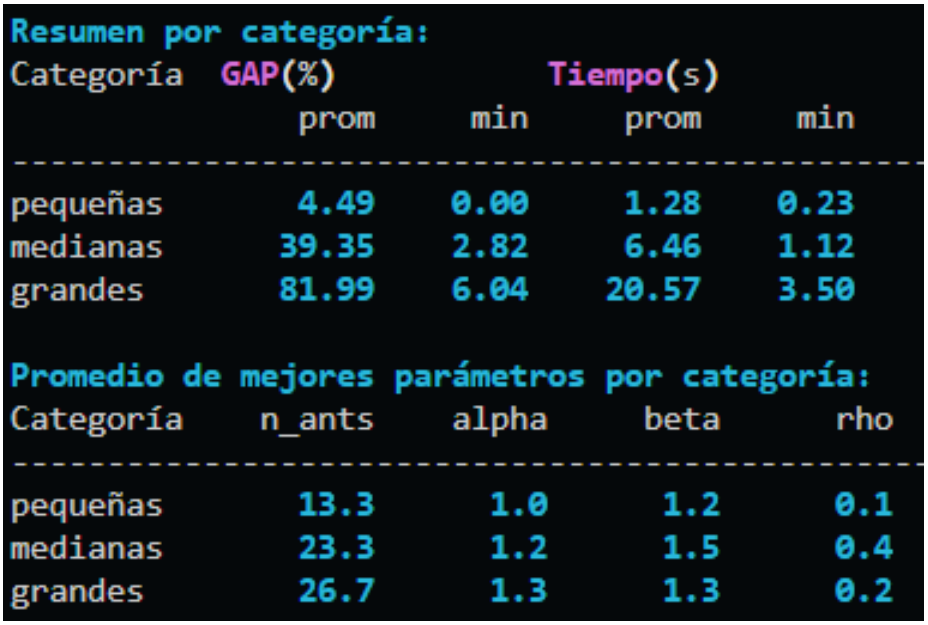
3.2.1 Resumen por categoría:

Categoría	GAP (%)		Tiempo (s)	
	Promedio	Mínimo	Promedio	Mínimo
Pequeñas	4.49	0.00	1.28	0.23
Medianas	39.35	2.82	6.46	1.12
Grandes	81.99	6.04	20.57	3.50

3.2.2 Promedio de Mejores Parámetros por Categoría

Categoría	n_ants	Alpha	Beta	Rho
Pequeñas	13.3	1.0	1.2	0.1
Medianas	23.3	1.2	1.5	0.4
Grandes	26.7	1.3	1.3	0.2

3.2.3 Captura de resultados de ejecución



4. Análisis

Instancias Pequeñas
- GAP promedio de 4.49%, con un mínimo de 0.00%
- Tiempo de ejecución promedio de 1.28 segundos
- Excelente rendimiento en burma14 (GAP 0.00%)
- Parámetros óptimos promedio: n_ants=13.3, α=1.0, β=1.2, ρ=0.1
- Alta eficiencia computacional con tiempos de convergencia rápidos

Instancias Medianas
- GAP promedio de 39.35%, con un mínimo de 2.82%
- Tiempo de ejecución promedio de 6.46 segundos
- Mejor rendimiento en eil51 y berlin52 (GAP 2.82%)
- Parámetros óptimos promedio: n_ants=23.3, α=1.2, β=1.5, ρ=0.4
- Incremento significativo en el GAP respecto a instancias pequeñas

Instancias Grandes
- GAP promedio de 81.99%, con un mínimo de 6.04%
- Tiempo de ejecución promedio de 28.57 segundos
- Mejor rendimiento en eil101 (GAP 6.04%)
- Parámetros óptimos promedio: n_ants=26.7, α=1.3, β=1.5, ρ=0.2
- Notable degradación del rendimiento en términos de GAP y tiempo

Influencia de los Parámetros
1. Número de hormigas (n_ants):
   - Incremento progresivo con el tamaño del problema
   - Pequeñas: 13.3 → Medianas: 23.3 → Grandes: 26.7
   - Mayor número de hormigas mejora la exploración pero aumenta el tiempo de cómputo

2. Alpha (α):
   - Tendencia ascendente con el tamaño del problema
   - Pequeñas: 1.0 → Medianas: 1.2 → Grandes: 1.3
   - Mayor importancia de las feromonas en problemas más complejos

3. Beta (β):
   - Relativamente estable entre categorías
   - Pequeñas: 1.2 → Medianas: 1.5 → Grandes: 1.5
   - Sugiere una importancia consistente de la información heurística

4. Rho (ρ):
   - Variación no lineal entre categorías
   - Pequeñas: 0.1 → Medianas: 0.4 → Grandes: 0.2
   - Indica la necesidad de ajuste específico según el tamaño del problema

5. Conclusiones

1. Escalabilidad:
  - El rendimiento del algoritmo ACO se degrada significativamente con el aumento del tamaño del problema
  - El GAP aumenta de 4.49% en instancias pequeñas a 81.99% en grandes
  - Los tiempos de ejecución se incrementan de manera no lineal

2. Efectividad:
  - Excelente para problemas pequeños (GAP < 5%)
  - Aceptable para problemas medianos (GAP ~ 39%)
  - Requiere mejoras para problemas grandes (GAP > 80%)

3. Parametrización:
  - Los parámetros óptimos varían según el tamaño del problema
  - La importancia de las feromonas (α) aumenta con el tamaño del problema
  - La influencia de la visibilidad (β) se mantiene relativamente constante

4. Recomendaciones:
  - Implementar técnicas de optimización local para mejorar soluciones
  - Considerar paralelización para instancias grandes
  - Desarrollar estrategias adaptativas de parametrización
  - Investigar hibridación con otros métodos metaheurísticos

5. Limitaciones y Trabajo Futuro:
  - El algoritmo actual muestra limitaciones en instancias grandes
  - Se requiere investigación adicional en:
    - Estrategias de inicialización de feromonas
    - Métodos de actualización adaptativa de parámetros
    - Técnicas de reducción del espacio de búsqueda
    - Paralelización y optimización del código

6. Implicaciones Prácticas

- El algoritmo es altamente recomendable para problemas TSP pequeños
- Para problemas medianos, se sugiere combinar con optimización local
- En problemas grandes, considerar otros algoritmos o versiones mejoradas del ACO
- La selección de parámetros debe adaptarse al tamaño del problema
- El compromiso entre calidad de solución y tiempo de ejecución debe evaluarse según el caso de uso específico

6. Código Fuente

El código fuente utilizado para llevar a cabo los experimentos se anexa en un enlace al repositorio en github.

https://github.com/JerikH/ACO