Algoritmos Exactos y Metaheurísticas

Primer Semestre 2025

Universidad Diego Portales
Prof. Víctor Reyes Rodríguez

Objetivos

- Algoritmo Tabu Search.
- Aplicaciones y ejemplos.

Clase anterior: Hill Climbing

- Es una técnica que permite ir mejorando una solución, generando en cada paso todo (o una parte) del vecindario.
- Cuando se llega a un óptimo local esta técnica incorpora el mecanismo de restart, con lo cual se comienza con una nueva solución y se busca tratando de mejorar la función objetivo.
- Una alternativa a recomenzar es permitir movimientos factibles que empeoren la solución actual, sin embargo podemos crear ciclos infinitos a menos que se prevenga la búsqueda de soluciones **repetidas**.
- Aunque este mecanismo de re-comenzar es eficiente, existen otros métodos que producen algoritmos más robustos. Entre ellos tenemos : Tabu Search y Simulated Annealing.
- Dificultad en la aceptación de **movimientos** que no mejoran la solución actual.

Tabu Search

- El objetivo de esta técnica es prevenir ciclos a corta duración, sin embargo, podrían re-visitarse soluciones sobre grandes intervalos de tiempo (iteraciones).

- Además de los ingredientes de Hill-Climbing (¿Cuáles?) tendremos una lista tabú.

- La lista tabú almacenará movimientos prohibidos y en cada iteración se elegirá el mejor movimiento factible no-tabú. Después de cada iteración, una colección de movimientos que podría hacer volver inmediatamente al punto anterior es agregado a la lista tabú. (memoria a corto plazo)

- Ya que a cada paso se puede mejorar o empeorar el valor de la función objetivo, se guarda la información de la mejor solución encontrada hasta el momento. Cuando la búsqueda termina se informa la mejor solución obtenida.

Tabu Search

- La lista tabú se actualiza normalmente en forma FIFO.

- El largo de la lista tabú controla la memoria del proceso de búsqueda (memoria a largo plazo). Una lista tabú corta, controla áreas reducidas del espacio de búsqueda y una larga, fuerza a una búsqueda en áreas mayores.

- El largo de la lista podría cambiar a lo largo del proceso de búsqueda.

- En este proceso se pierde información, y buenas soluciones pueden ser excluidas del conjunto permitido. Para reducir este problema, se define un **criterio de aspiración** que permitiría a una solución estar dentro del conjunto de soluciones permitidas aún cuando figure en la lista tabú.

Elementos claves

- Restricciones Tabú: Restringir la búsqueda al clasificar ciertos movimientos como prohibidos (tabú), para evitar caer en soluciones recientemente generadas.

- Criterio de aspiración: Liberar la búsqueda por medio de una función de memoria a corto plazo (olvido estratégico).

Pseudocódigo TS.

```
Proceso Tabu Search
    sol-actual = Inicialización
    lista-tabú = Vacía
    Mientras (No se cumpla el criterio de parada) Hacer
        Seleccionar la mejor solución no tabú del vecindario y almacenar como
        sol-actual, actualizar lista tabú
        Si (fo(sol-actual) es mejor que fo(mejor-sol))
            mejor-sol = sol-actual
    Fin Mientras
Fin Proceso Tabu Search
```

Exploración y explotación en TS

- Explotación/Intensificación: Búsqueda local (elementos de Hill-Climbing).

- Exploración/Diversificación: A través de la lista tabú.

- Balance estático: a través del tamaño de la lista tabú.
  - Lista tabú larga: gran diversificación y poca intensificación.
  - Lista tabú corta: gran intensificación y poca diversificación.

- Cambios dinámicos a lo largo de la lista tabú (Reactive Tabu Search). El largo de la lista varía según las propiedades de la trayectoria en el espacio de búsqueda.

Ejemplo

- Supongamos que queremos optimizar la disposición de cierto conjunto de tareas (digamos siete) de tal manera de maximizar la calidad del trabajo realizado.
- Además supondremos que la función objetivo es una caja negra (sólo con el propósito del ejemplo)
- Representación: La solución estará dada por un vector de tamaño 7 en donde en cada casilla se ubicará el identificador de la tarea.
- Operador de vecindario: El operador swap. Se intercambia dos elementos de posición. Por ejemplo:
  - (2,5,7,3,5,6,1) → (2,6,7,3,4,5,1)
- Tam lista tabú: 3

Ejemplo

- **Iteración 0**: Sol (2,5,7,3,4,6,1) f.o=10 ; Lista tabú: vacía ; Vecindario (mejores 5): (5,4) → +6, (7,4) → +4, (3,6) → +2, (2,3) → +0, (4,1) → -1.

- **Iteración 1**: Sol (2,4,7,3,5,6,1) f.o=16; Lista tabú: (5,4) ; Vecindario (mejores 5): (3,1) → +2, (2,3) → +1, (3,6) → -1, (7,1) → -2, (6,1) → -4.

- **Iteración 2**: Sol (2,4,7,1,5,6,3) f.o=18; Lista tabú: (5,4),(3,1) ; Vecindario (mejores 5): (1,3) → -2, (2,4) → -4, (7,6) → -6, (4,5) → -7, (5,3) → -9.

Ejemplo

- **Iteración 3**: Sol (4,2,7,1,5,6,3) f.o=14; Lista tabú: (5,4),(3,1),(2,4) ; Vecindario (mejores 5): (4,5) → +6, (5,3) → +2, (7,1) → +0, (1,3) → -3, (2,6) → -6.

- **Iteración 4**: Sol (5,2,7,1,4,6,3) f.o=20; Lista tabú: (3,1),(2,4),(5,4) ; Vecindario (mejores 5): → (7,1) +0, (4,3) → -3, (6,3) → -5, (5,4) → -6, (2,6) → -8.

Resumen

- Estudiamos Tabu Search y sus variantes.
- Próxima clase : Simulated annealing (última MH de trayectoria)