Inteligencia Artificial
Progressive Party Problem Especificaciones

Camila Díaz
cdiaz@alumnos.inf.utfsm.cl

10 de junio de 2014

1. Objetivo del Problema

- Maximizar la cantidad de bloques de duración de la fiesta comenzando desde T = 6.

2. Restricciones del Problema

1. Un bote puede ser visitado solo si es un bote anfitrión.
2. La capacidad de un bote anfitrión no puede ser superada.
3. Cada tripulación invitada debe siempre tener un anfitrión asociado.
4. Una tripulación invitada no puede visitar el mismo bote anfitrión más de una vez.
5. Cualquier par de invitados debe encontrarse a lo más una vez.

3. Datos de Entrada e Instancias de Prueba

Los datos iniciales debes ser entregados al programa mediante la lectura de los archivos de prueba. Los archivos de entrada serán los siguientes:

1. Datos:
   - ppp.cap : Contiene las capacidades (columna 1) y tripulantes (columna 2) de cada bote por fila. Éste será el archivo base a utilizar. [1]
2. Instancias: Contienen un booleano indicando si el barco (fila) es anfitrión (1) o visitante (0).
   - ppp_1.hst : Instancia original botes anfitriones 1-13.
   - ppp_2.hst : Instancia botes anfitriones 1-2,16.
   - ppp_3.hst : Instancia botes anfitriones 1,3-13,19.
   - ppp_4.hst : Instancia botes anfitriones 3-13,25,26.
   - ppp_5.hst : Instancia botes anfitriones 1-11,19,21.
   - ppp_6.hst : Instancia botes anfitriones 1-9,16-19.

Los archivos estarán disponibles en `labcomp.cl/~cdiaz/ppp_data.tar.gz`.

[1] Considerar que de todas maneras se debe generar un algoritmo genérico que permita la entrada de otros archivos de datos que puedan tener más o menos de 42 botes.

4. Salida del Programa

La salida del programa debe ser un archivo *.sol, que contenga la calendarización de cada barco por cada periodo de tiempo. Considerar que para cada instancia se deberan generar más de un archivo de salida, uno por cada T encontrado.

Un archivo (ppp-ref_1_6.sol) correspondiente a una de las soluciones encontradas para la instancia original con 6 periodos será entregado como referencia de un archivo de salida.

5. Recomendaciones

- Comentar adecuadamente el código.
- La solución **debe ser implementada en C/C++ en entorno Linux**.
- Definir los parámetros desde un principio con el fín de lograr la mejor sintonización de estos (en el informe final se pide conclusiones sobre los experimentos realizados para ello).
- Redactar correctamente el archivo README especificando la forma en que debe ejecutarse el código con los archivos de entrada.
- Recordar que para el análisis de resultados es recomendable obtener datos extras de la implementación, como tiempo de ejecución, movimientos, sintonizaciones del código, entre otros.