Universidad Técnica Federico Santa María
Departamento de Informática

Instrucciones Entregable #2 - MRP

Inteligencia Artificial

Ayudante: Darío Canales

dario.canales@alumnos.usm.cl

Junio 2015

1. Problema, objetivos y restricciones

El problema a resolver será el Machine Reassignment Problem definido en [1]. La idea principal es que luego de implementar su algoritmo puedan comparar sus resultados con los de la literatura.

1.1. Objetivo del problema

El objetivo del problema es encontrar una reasignación de las máquinas que cumpla con todas las restricciones y minimice la función de costos dada.

1.2. Restricciones

Se deben respetar las restricciones de capacidad, conflictos, dispersión, dependencia y uso recursos transitorios, definidos en la definición del problema ([1]).

2. Instancias de prueba

Las instancias de prueba que se utilizarán serán las provistas en [1][1], las cuales son 20 en total y están divididas en tres grupos en orden creciente de dificultad. Cada instancia consta de dos archivos, uno que contiene la asignación inicial y otro con los demás datos, las características generales de cada instancia se presentan a continuación.

[1] Sección "Instances and Checker" del sitio web

Conjunto de instancias	Instancia	N° Recursos	N° Máquinas	N° Servicios	N° Procesos
A1	1	2	4	79	100
A1	2	4	100	980	1000
A1	3	3	100	216	1000
A1	4	3	50	142	1000
A1	5	4	12	981	1000
A2	1	3	100	1000,	1000
A2	2	12	100	170	1000
A2	3	12	100	129	1000
A2	4	12	50	180	1000
A2	5	12	50	153	1000
B	1	12	100	2512	5000
B	2	12	100	2462	5000
B	3	6	10	15025	20000
B	4	6	500	1732	20000
B	5	6	100	35082	40000
B	6	6	200	14680	40000
B	7	6	4000	15050	40000
B	8	3	100	45030	50000
B	9	3	1000	4609	50000
B	10	3	5000	4896	50000

Figura 1: Datos de las instancias

2.1. Formato de instancias de prueba

El formato de instancias de pruebas esta especificado en detalle en [1] en la definición del problema.

3. Especificaciones del input del programa

El programa será llamado de la siguiente manera:
./mrp -t time_limit -p instance_filename -i original_solution_filename -o new_solution_filename -s seed
Donde cada parámetro significa lo siguiente:

1. mrp: es el nombre del programa
2. -t time_limit: Especifica el tiempo límite antes de terminar el programa donde "time_limit" es la cantidad de segundos (número entero), luego de transcurrido este tiempo el programa **debe** arrojar los output especificados más adelante. En caso de las técnicas completas el parámetro "time_limit" podría ser 0, lo cual significa que el programa no se detendrá hasta encontrar el óptimo global.
3. -p instance_filename: Especifica el nombre del archivo de instancia, donde "instance_filename" es el nombre del archivo con extensión, el cual estará en la misma carpeta del programa.
4. -i original_solution_filename: Especifica el archivo de asignación inicial, donde "original_solution_filename" es el nombre del archivo con extensión, el cual estará en la misma carpeta del programa.

5. -o new_solution_filename: Especifica el archivo de resultados, donde "new_solution_filename" es el nombre del archivo con extensión, el cual deberá ser escrito en la misma carpeta del programa.

6. -s seed: Especifica la semilla utilizada para generar números aleatorios, donde "seed" es un número entero. Esto servirá para poder generar secuencias de números aleatorios dependientes de los inputs del programa y así su investigación sea reproducible([2]).

   Ejemplo de ejecución:

   mrp -t 300 -p model_a1_1.txt -i assignment_a1_1.txt -o salida.txt -s 100

Usted puede añadir otros parámetros adicionales en la entrada dependiendo de su técnica, tal como cantidad de iteraciones máxima, temperatura, cantidad de restarts, etc. Si lo hace, **debe incorporarlos junto con una descripción en el archivo README.txt**, estableciendo al menos un ejemplo de cómo ejecutar su programa con estos parámetros.

4. Especificaciones del output del programa

Su programa debe tener dos ouputs luego de terminar su ejecución.

1. Archivo de texto cuyo nombre fue especificado en la línea de parámetros: el cual contendrá cada una de las asignaciones, tal como se especifica en la definición del problema de [1]. **Notar que en el sitio existe un verificador de soluciones, el cual puede ser utilizado para archivos de este tipo**[2].

2. Salida por pantalla, la cual contendrá los valores separados cada uno por un ";" (punto y coma) de cada uno de los siguientes datos de ejecución:

   a) El valor de la función objetivo.

   b) Tiempo de ejecución en segundos (el cual podría ser distinto del tiempo límite, en caso de que su programa finalice antes).

   c) Cantidad de iteraciones alcanzada (en caso de una técnica incompleta). Cantidad de instanciaciones alcanzada (en el caso de una técnica completa).

   Ejemplo de salida por pantalla:

   10;80;500 : lo cual significa que el valor de la función objetivo es 10, el programa se ejecutó por 80 segundos antes de terminar y la cantidad de iteraciones fue 500.

   Este output servirá para ver el resumen de ejecución de su programa.

Si desea mostrar otros datos que considere relevantes (y que puedan visualizarse fácilmente) puede añadirlos al final del output por pantalla, **especificando en el archivo README.txt su significado y qué lo motivó a añadirlos a la salida del programa**.

[2] Ruta directa: http://challenge.roadef.org/2012/files/solution_checker/

5. Recomendaciones y recordatorios

1. Recuerde que su programa debe ser realizado en C/C++ con un Makefile para poder compilarlo.
2. Recuerde que debe COMENTAR su código. No olvide además poner nombres de variables, funciones, clases y métodos que tengan relación con su utilización. Ej. (de función) void comprobarRestricciones().
3. Recuerde incluir un README donde se especifiquen:
   a) Instrucciones de ejecución.
   b) Parámetros necesarios para ejecutar el algoritmo, con una explicación de cada uno (significado, objetivo del parámetro, etc.).
   c) Ejemplo(s) de comando(s) para ejecutar el algoritmo.
   d) En caso de requerirlo, el detalle de datos adicionales incluidos en el output.
   e) Cualquier supuesto que considere relevante.
4. Si tiene problemas con algunas instancias específicas producto de, por ejemplo, cómputos muy grandes o demasiada memoria RAM utilizada que saturó su pc, se debería hacer lo siguiente:
   a) Explicar en el informe cuáles instancias fueron las que originaron el problema
   b) Explicar cuál fue el problema que se detectó: tamaño de la instancia muy grande, error aleatorio, etc.
   c) En caso de una técnica completa, si su programa o pc se queda pegado y no puede terminar la ejecución del programa correctamente, se recomienda añadir al código algún mecanismo que permita guardar periódicamente los dos outputs en archivos, el primero con la reasignación parcial hasta ese momento y el segundo con los parámetros pedidos separados por punto y coma.
5. Si bien no se han especificado valores óptimos para las instancias entregadas que permitan finalizar su programa antes del tiempo límite, para las instancias del conjunto A, usted puede establecer como condiciones de término de su algoritmo, valores iguales o semejantes a las soluciones provistas en [1][3], las cuales corresponden a las mejores soluciones para cada una de las instancias en la fase de calificación.
6. **Recuerde que el segundo entregable debe contener tres cosas: el CD con el código y el informe realizado en digital, el Informe 2 impreso y el Informe 1 con las correcciones solicitadas (este último es la misma impresión que entregó en la primera ocasión, para poder revisar si se realizaron los cambios pedidos).**

[3] Sección "Qualification results"

Referencias

[1] Google roadef/euro challenge 2011-2012: Machine reassignment problem. `http://challenge.roadef.org/2012/en/`. Accedido: 01-06-2015.

[2] Matthias Schwab, Martin Karrenbach, and Jon Claerbout. Making scientific computations reproducible. *Computing in Science & Engineering*, 2(6):61–67, 2000.