Rotative Workforce Scheduling Problem

Presentación Final

Jonathan Antognini C.
Luis Casanova S.

Universidad Técnica Federico Santa María

24 de julio de 2012

Introducción

Estructura de datos

Los datos se obtuvieron de los input (Example*.txt), archivos que contenían los datos necesarios para poder definir el problema. Los datos son:

- w: largo de la planificación. Todos los input tenían un $w = 7$, lo que corresponde a planificar de lunes a domingo.
- n: número de empleados.
- m: número de turnos más día libre. En la mayoría de los input m valía 4 (3 turnos + 1 día libre).
- A: vector de largo m donde se indican los diferentes tipos de turnos. En la mayoría de los casos v contenía $D, A, N, -$.

- R: matriz de requerimientos de turnos por día, ya que hay w días, y $m-1$ día son turnos, la matriz tiene dimensión $R_{(m-1)xw}$
- $MAXS$: Vector de largo m donde por cada turno se indica el máximo de turnos o días libres consecutivos permitidos.
- $MINS$: Vector de largo m donde por cada turno se indica el mínimo de turnos o días libres consecutivos permitidos.
- $MAXW$: número máximo de días consecutivos trabajados.
- $MINW$: número mínimo de días consecutivos trabajados.
- $C2$: matriz con secuencias de turnos no permitidas de largo 2.
- $C3$: matriz con secuencias de turnos no permitidas de largo 3.

Foward Checking

Graph BackJumping

Representación para ambos algoritmos

Considerando $w = 4$ y $n = 3$:

Lu	Ma	Mi	Ju
A	A	D	-
D	D	N	N
-	-	A	N

Para calcular efecto de comprobación de restricciones se representó de la siguiente forma:

Lu	Ma	Mi	Ju	Lu	Ma	Mi	Ju	Lu	Ma	Mi	Ju
A	A	D	-	D	D	N	N	-	-	A	N

Greedy

Para poder definir un algoritmo Greedy correctamente es necesario especificar:

- Función de evaluación: esta función es la misma definida en la sección anterior.
- Punto de partida: el día en donde se quiera empezar a planificar. Es decir se le entregará un día entre 1 y W.
- Función miope: para el día i, se le asigna al primer trabajador disponible el turno requerido, de tal forma que la diferencia entre la cantidad de empleados necesarios en dicho turno se minimize.

Hill Climbing

- Número de restart: definido como constante (se cambiaba por cada instancia).
- Solución inicial: solución obtenida mediante greedy.
- Función objetivo: la función objetivo corresponde a minimizar la cantidad de penalizaciones hechas debido a restricciones blandas insatisfechas.

- Movimiento: swaps de turnos entre turnos de un día. Por ejemplo:

Lu	Ma	Mi	Ju
A	A	D	-
D	D	N	N
-	-	A	N

Al aplicar el movimiento y generar el primer vecino, se hace un swap de la casilla 1,1, con la casilla 2,1, quedando:

Lu	Ma	Mi	Ju
D	A	D	-
A	D	N	N
-	-	A	N

Greedy + HC

- Se inicializa una solución vacía.
- Se le pasas esa solución vacía al greedy, y además un día de comienzo. El resultado de esta operación genera una solución que respeta las restricciones duras (R).
- La solución del greedy es la entrada ahora para el hill climbing.
- Se realiza el movimiento sobre la solución actual.
- Si el algoritmo no encuentra un mejor vecino, entonces se hace un restart.
- Cuando se termina la cantidad de restart, el algoritmo acaba y entrega la mejor solución encontrada.

Resultados

Los resultados encontrados mediante greedy+hc fueron:

Instancia	Valor función objetivo
Example1.txt	35
Example2.txt	52
Example3.txt	79
Example4.txt	32
Example5.txt	46
Example6.txt	22
Example7.txt	289
Example8.txt	71

Conclusiones