Investigación Operativa - Trabajo Práctico

29 de Octubre de 2012

1 Introducción

Los problemas de Optimización Combinatoria aparecen en muchas situaciones y disciplinas de la vida real, desde la Genética, la Física y la Química a las Finanzas y la industria en general. La mayoría de estos problemas es fácil de modelar matemáticamente pero difíciles de resolver computacionalmente. Una forma de formular este tipo de problemas es mediante modelos de Programación Lineal Entera Mixta (PLEM), donde lo que buscamos es resolver un problema de Programación Lineal en el cuál un subconjunto de sus variables son definidas como enteras.

Durante las últimas décadas, la utilización de modelos PLEM se ha convertido en una de las técnicas más efectivas para resolver problemas difíciles de Optimización Combinatoria. En térmios generales, podemos escribir un modelo de PEM de la siguiente forma

$$\begin{aligned}
\min \quad & \sum_{j \in I} c_j x_j + \sum_{j \in C} c_j x_j \\
\text{s.t.} \quad & \sum_{j \in I} a_{ij} x_j + \sum_{j \in C} a_{ij} x_j \geq b_i, \quad i = 1, \ldots, m \\
& x_j \in \mathbb{Z}_+ \quad \forall\ j \in I \\
& x_j \in \mathbb{R}_+ \quad \forall\ j \in C
\end{aligned}$$

donde I es el conjunto de variables enteras y C el conjunto de variables continuas, con $I \cup C = \{1, \ldots, n\}$.

Los problemas de PLEM en su forma general pertencen a la clase $\mathcal{NP}$-hard. Sin embargo, los algoritmos exactos de tipo *Branch and Bound* y *Branch and Cut* suelen ser muy efectivos en la práctica. Esto se debe principalmente a la investigación de técnicas algoritmcas y el desarrollo teórico realizados durante los últimos años, con el objetivo de obtener de tanto algoritmos de propósito general como el estudio de modelos de PLEM en particular.

2 Enunciado

El objetivo de este trabajo es desarrollar y evaluar los distintos tipos de algoritmos para problemas PLEM vistos en la segunda mitad de la materia para problemas de Programación Binaria (PB), un caso particular de problemas de PLEM donde todas las variables

pertenecen a $\{0, 1\}$, sobre instancias de *benchmark* utilizadas en la literatura. El trabajo será en grupos de 2 o 3 personas como máximo.

Los algoritmos a implementar son:

1. Algoritmo *Branch and Bound*
2. Algoritmo *Cut and Branch*
3. Algoritmo *Branch and Cut*

Para el desarrollo de los algoritmos se utilizarán las librerías que provee el paquete CPLEX, preferentemente la *Callable Library*.

Los algoritmos *Cut and Branch* y *Branch and Cut* requieren el desarrollo de algoritmos de planos de corte, que deberán incluir las siguientes desigualdades válidas:

1. Cortes *Cover* o cortes de *Gomory*.
2. Cortes Clique.

Para los Cortes Clique, el objetivo será generar un grafo de conflictos utilizando como base las técnicas propuestas en [1]. Además, para cada uno de los tipos de cortes implementados será necesario diseñar e implementar rutinas (eventualmente heurísticas) de separación.

Los distintos algoritmos dispondrán de un tiempo máximo de ejecución y serán comparados y evaluados en función de distintos parámetros para cada instancia:

1. tiempo de resolución,
2. cantidad de nodos generados,
3. %gap en el nodo raíz,
4. en caso de haber alcanzado el tiempo límite, el %gap alcanzado.

Con el fin de realizar una comparación justa entre los algoritmos y poder identificar cuál es el impacto de las distintas técnicas, los algoritmos deberán tener deshabilitados todos los cortes, heurísticas y preprocesamientos provistos por CPLEX. Además, se deberán fijar las estrategias de branching y de selección de nodos. Finalmente, los algoritmos serán evaluados sobre un conjunto adecuado de intancias de problemas de PB, que serán tomados de [2].

Fecha de entrega: Lunes 10 de Diciembre de 2012

References

[1] Alper Atamtürk, George L. Nemhauser, and Martin W. P. Savelsbergh. Conflict graphs in solving integer programming problems. *European Journal of Operational Research*, 121(1):40–55, 2000.

[2] Thorsten Koch, Tobias Achterberg, Erling Andersen, Oliver Bastert, Timo Berthold, Robert E. Bixby, Emilie Danna, Gerald Gamrath, Ambros M. Gleixner, Stefan Heinz, Andrea Lodi, Hans Mittelmann, Ted Ralphs, Domenico Salvagnin, Daniel E. Steffy, and Kati Wolter. MIPLIB 2010. *Mathematical Programming Computation*, 3(2):103–163, 2011.