Proyecto 2 Flujo en Redes

Nicolás Hernández1, Nicolás Vásquez2

1 Universidad de los Andes  
[nm.hernandez10@uniandes.edu.co](mailto:nm.hernandez10@uniandes.edu.co)

2 Universidad de los Andes  
[n.vasquez10@uniandes.edu.co](mailto:n.vasquez10@uniandes.edu.co)

Abstract

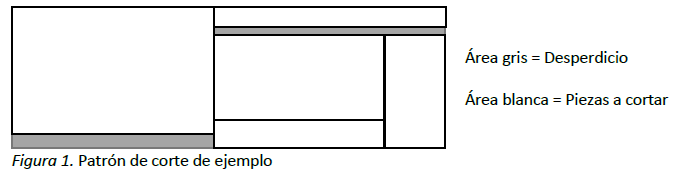
Este documento contiene el desarrollo, explicación, implementación y los resultados del modelo matemático formulado para encontrar el patrón de corte que brinde el menor tiempo posible y cuya implementación obtenga un tiempo computacional bajo. Se realizó una implementación de la formulación de un problema CPP (Chinese Postman Problem) modificado para adecuarlo al problema de cortes utilizando el optimizador Gurobi con el API de Java.

# Introducción

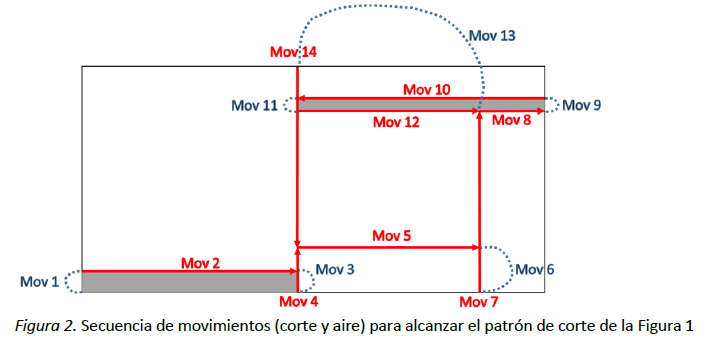
Este documento contiene el desarrollo, explicación, implementación y los resultados del modelo matemático formulado para encontrar el patrón de corte que brinde el menor tiempo posible y cuya implementación obtenga un tiempo computacional bajo. Se realizó una implementación de la formulación de un problema CPP (Chinese Postman Problem) modificado para adecuarlo al problema de cortes utilizando el optimizador Gurobi con el API de Java.

## Descripción del problema

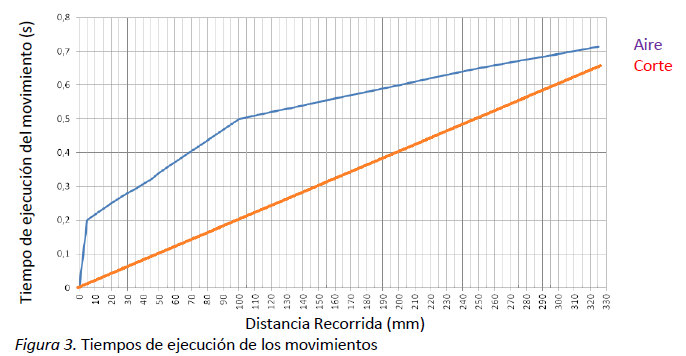
Implementación, experimentación, análisis de resultados obtenidos y elaboración de informe de una metodología de optimización para el problema de secuenciamiento de corte bidimensional de material.



Existen diferentes tipos de movimientos dependiendo de la tecnología de corte, en el problema de corte simulado existen solo dos tipos: movimiento de corte y movimiento en el aire. El primero consiste en un movimiento durante el cual el efector final (marcador) viaja dejando huella en la placa de material (mesa de trabajo). Por otro lado, el movimiento en el aire sucede cuando el cortador (brazo robótico) se desplaza de un lugar a otro sin dejar huella (ver Figura 2).



Ambos movimientos tienen un consumo de tiempo diferente, en especial ambos comportamientos son no lineales, para facilidad de calcular los tiempos de ejecución de los movimientos asuma los tiempos ilustrados en la Figura 3.



El objetivo de problema de secuenciamiento de corte es minimizar el tiempo total para llevar a cabo la tarea de corte, el tiempo total está compuesto por la suma de los tiempos de los movimientos más el tiempo de cómputo requerido por la metodología de solución.

Cabe resaltar que cortes contiguos entre piezas se pueden realizar una única vez, mientras que cortes contiguos con los límites de la placa no son necesarios de realizarlos (ver Figura 2).

La estructura del Proyecto 1 se espera que conserve un diseño metodológico (en formato artículo, anexo en este anuncio), donde se especifique claramente la descripción del problema, el esquema de optimización (modelo matemático, pseudocódigo, etc.), junto a las ventajas y desventajas de este. Además de esto, se debe implementar la metodología (en cualquier lenguaje de programación computacional o matemático) y se debe realizar la experimentación y validación del desarrollo, a través del uso de los patrones de corte obtenidos por (Cuellar-Usaquén, 2018) para las 17 instancias (GCUT1-17) propuestas por (Beasley, 1985) contenidas en el archivo comprimido PatronesCorte.rar (adjunto en este anuncio). Sumado a la experimentación se debe adicionar un análisis de los resultados obtenidos con relación a las soluciones alcanzadas.

## Esquema de optimización

Para resolver este problema, se va a realizar un grafo, donde los nodos representen las esquinas de las figuras que se cortan y los arcos el movimiento de la máquina (ya sea en corte o sin realizar corte). Luego, para resolver el problema, se formuló la solución como un problema de optimización lineal entera, donde se utilice la estructura de un modelo clásico de flujo en redes solucionando un problema de ruta más corta, añadiendo una restricción que obligue a que se visiten los arcos que corten las piezas y que solamente se pase por un corte una sola vez.

# Metodología

## Realización del grafo

Para realizar el grafo, se carga el archivo de texto que contiene las coordenadas de las figuras a cortar. En primer lugar, todas estas esquinas son un nodo en el grafo, así que se agregan todos los nuevos nodos sin alguna. Luego, para todo nodo existente se crea un nodo “gemelo” que represente la misma esquina en la misma coordenada. Esto, se hace con el motivo de representar que hay dos maneras de llegar a una esquina: una es por aire y otra es cortando. Adicionalmente, se crean dos nodos cuyo objetivo es ser el nodo inicial y nodo final para el problema de “ruta más corta” y no representan ninguna coordenada.

Luego, se crean los arcos entre los nodos. Primero, debe haber un arco entre todos los nodos de aire, puesto que existe un movimiento entre todas las esquinas del patrón que se realice por aire. Además, el costo vinculado con este arco será igual al tiempo que tome entre las coordenadas del nodo incial y del nodo final viajando por aire de acuerdo con los tiempos de la Figura 3. Por otro lado, todos los nodos corte, tienen un arco que los conecta con la esquina más cercana a la cual se pueda llegar por medio de un corte factible, es decir, sin cortar por espacios donde se arruinen las figuras. Así mismo, el costo vinculado a este arco, será el tiempo que tome moverse entre ambas esquinas haciendo un corte. Por lo tanto, este arco representa realizar un corte entre las dos coordenadas (inicial y final) del arco.

Adicionalmente, se crean arcos con un costo vinculado de 0, dado que es una transición entre dos nodos que no implica un movimiento en coordenadas de la máquina de corte. Estos arcos, son los que conectan cada nodo de corte con su respectivo nodo de aire, dado que esto no representa ningún movimiento para la máquina, sino una transición necesaria para identificar si va a realizar un corte o si se va a mover por el aire. Además, existe un arco entre el nodo inicio y todos los nodos que no sean el nodo final, así como hay un arco entre todos los nodos y el nodo final excepto por el nodo inicio. Estos arcos tampoco tienen ningún costo (tiempo) asociado. Por último, cabe recalcar que todos los arcos son no dirigidos, y por lo tanto se crean dos arcos dirigidos por cada uno (uno de ida y uno de vuelta).

## Formulación del problema

Se realizó una formulación lineal entera para solucionar el problema de corte como un problema de flujo en redes de ruta más corta, utilizando los nodos inicio y final explicados en la realización del grafo, añadiendo una restricción para que se cortaran los arcos de corte una sola vez y llegar a la solución para cada instancia. La formulación es la siguiente:

**Conjuntos:** Se cuenta con un conjunto de arcos total que representan los movimientos de la máquina de corte, un subconjunto de arcos de corte y un conjunto de nodos que representa las esquinas de las figuras que se deben cortar.

**Variables de decisión:** Se crea una variable de decisión binaria para cada arco en el problema, que toma el valor de 1 si el arco hace parte de la solución y 0 de lo contrario, es decir, cuando el arco se utiliza la máquina se va a mover entre las coordenadas de la esquina y la .

**Parámetros:** Como parámetro se obtiene el tiempo de movimiento para todos los arcos del problema.

**Función objetivo:** Se debe minimizar el tiempo total de corte:

(1)

**Restricciones:**

1. Restricción de balance:

(2.1)

(2.2)

(2.3)

1. Restricción para pasar obligatoriamente una sola vez por los arcos de corte:

(3)

1. Naturaleza de las variables

(4)

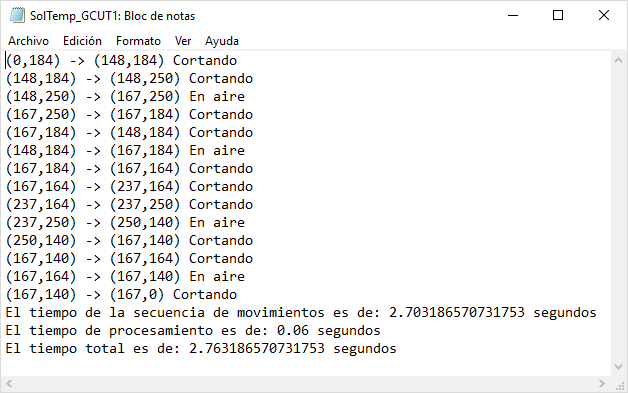
# Implementación Java + Gurobi

Se utilizó el optimizador Gurobi para resolver el problema formulado, utilizando el API de Java. La aplicación ejecuta una clase llamada Model.java. Esta, primero carga el archivo de datos del patrón, arma el grafo en la clase Reader.java y genera la matriz de adyacencias. Luego, desde la clase Model.java se construye el modelo a optimizar: se crean las variables, restricciones y el sentido del problema (maximización o minimización). Luego de esto, resuelve el modelo y obtiene los arcos que hacen parte de la solución óptima. Por último, ordena la solución como un solo camino para generar un archivo de texto con las coordenadas que debe visitar la máquina en orden indicando si el movimiento es de corte o por el aire.

# Resultados

## Resultados para las instancias proporcionadas

Para todas las instancias, se genera un camino óptimo ordenado a seguir. Estos resultados se imprimieron en un archivo para cada instancia. Por ejemplo, para la primera instancia de corte se genera el archivo que muestra la Figura 4.

  
**Figura No. 4:** Resultados para instancia 1 de corte

A continuación, presentamos un cuadro comparativo mostrando el tiempo computacional que se demora cada patrón, el número de arcos del grafo original, el número de nodos y el número de arcos que componen el camino óptimo que debe tomar. Además, se muestra el tiempo total de solucionar el problema y el tiempo que toma el camino en ser cortado para componer el tiempo total de cada patrón de corte.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Patron** | **Arcos** | **Nodos** | **Arcos Final** | **Solución**  **óptima (seg)** | **Tiempo**  **procesamiento (seg)** | **Tiempo total (seg)** |
| **1** | 196 | 24 | 14 | 2.70318 | 0.046 | 2.74918 |
| **2** | 644 | 46 | 29 | 4.43334 | 0.080 | 4.51334 |
| **3** | 290 | 30 | 14 | 3.40402 | 0.048 | 3.45202 |
| **4** | 498 | 40 | 23 | 4.56981 | 0.071 | 4.64081 |
| **5** | 168 | 22 | 9 | 5.76200 | 0.046 | 5.80800 |
| **6** | 292 | 30 | 16 | 6.70921 | 0.061 | 6.77021 |
| **7** | 166 | 22 | 10 | 4.69618 | 0.059 | 4.75518 |
| **8** | 224 | 26 | 13 | 4.94554 | 0.050 | 4.99554 |
| **9** | 200 | 24 | 13 | 12.22400 | 0.045 | 12.26900 |
| **10** | 194 | 24 | 12 | 9.23307 | 0.050 | 9.28307 |
| **11** | 326 | 32 | 18 | 12.06041 | 0.059 | 12.11941 |
| **12** | 194 | 24 | 12 | 9.26453 | 0.040 | 9.30453 |
| **13** | 252989 | 312 | 212 | 151.14153 | 1.053 | 152.19453 |
| **14** | 7176 | 164 | 104 | 95.42805 | 0.410 | 95.83805 |
| **15** | 7166 | 164 | 102 | 103.28269 | 0.812 | 104.09469 |
| **16** | 9532 | 190 | 128 | 110.48058 | 0.704 | 111.18458 |
| **17** | 4562 | 130 | 90 | 86.10935 | 0.563 | 86.67235 |

**Tabla No. 1:** Reporte de tiempos de ejecución

# Análisis de resultados

Al ejecutar la solución para cada una de las instancias de corte proporcionadas, se encontraron similitudes entre el tamaño del grafo y el tiempo total de ejecución. Además, se compararon las características de los modelos resueltos para reconocer cuál era más complejo y cuál es la solución que implica un mayor tiempo computacional.

En primer lugar, se compara el tamaño del grafo con los tiempos reportados de la solución. En la primera gráfica, se compara el número de nodos del grafo construido con el tiempo que se demora en construir el grafo, armar y resolver el modelo de optimización, organizar el camino de la solución y reportar los resultados. En esta gráfica, es notable que a mayor cantidad de nodos (es decir, mientras más piezas haya que cortar), el tiempo que se demora en encontrar el camino óptimo es mayor. Así mismo, en la Gráfica No. 2, se compara el tamaño del grafo con el tiempo que toma el corte en la solución óptima. En esta gráfica, también se puede observar que a medida que aumenta el tamaño del grafo, es mayor el tiempo que se toma en cortar las piezas.

**Gráfica No. 1:** Número de nodos vs. Tiempo procesamiento

**Gráfica No. 2:** Número de nodos vs. Tiempo de corte total

Por último, se compararon los tiempos totales para cada una de las instancias suministradas. La instancia que toma un mayor tiempo total es el patrón de corte 13, y el que toma un menor tiempo es la instancia 1. En la gráfica No. 2, se demuestra que los patrones que tienen una mayor cantidad de nodos, son aquellos que presentan un mayor tiempo total.

**Gráfica No. 3:** Tiempo total para cada patrón

**Gráfica No. 4:** Número de nodos total por patrón de corte.

# Conclusiones

Finalmente, es posible formular y resolver muchos problemas de la vida real por medio de la estructura de datos de un grafo, ya sea ruteo en arcos o ruteo en nodos. En cuanto a los ruteos en arcos, existen muchos métodos para resolver de manera óptima los modelos. Es posible, también, adaptarlos a un problema de ruta más corta (problema de ruteo en nodos) y resolverlo con una formulación de flujo en redes adaptando una restricción adicional para que recorra los arcos necesarios.

Adicionalmente, concluimos que a medida que crece el tamaño de un grafo, crece la complejidad computacional (y por lo tanto el tiempo computacional) de resolver el problema. Esto, sucede dado que los grafos como estructuras de datos son complejos y son utilizados para resolver problema de tipo NP Hard, lo cual implica que los algoritmos implementados, aumentan de manera no-polinomial a medida que aumenta el tamaño del mismo. Por lo tanto, es muy importante para este tipo de problemas encontrar soluciones que crezcan en complejidad en el menor orden posible.

References

1. James Brown. Why the work of John Smith must not be cited. Personal communication, 2011.
2. James Brown and John Smith. How not to cite papers. In John Smith, editor, *Proceedings of the First International Conference on Modern Bibliometrics (MODBIB 2009*), pages 20–30, Pasadena (CA), USA, July 25–28 2009.
3. John Smith. How to make citations. *Journal of Modern Bibliometrics*, 1:1–10, 2010.
4. John Smith. Against the accusations of James Brown. Technical Report 01-11, CS Department, University of Serendipity, 2011.