# Soluciones óptimas del Master Surgical Schedule (MSS)

1st Kevin Arturo Amaya Osorio
*dept. ciencias de la computación*
*UNAL*
Bogotá D.C., Colombia
kaamayao@unal.edu.co

2nd David Ernesto Ramirez Arboleda
*dept. ciencias de la computación*
*UNAL*
Bogotá D.C., Colombia
deramireza@unal.edu.co

3th Iván Gabriel Aranguren Rengifo
*dept. ciencias de la computación*
*UNAL*
Bogotá D.C., Colombia
iaranguren@unal.edu.co

***Resumen*—Uno de los recursos más vitales y limitados en los hospitales es el tiempo del uso de quirófanos y sala de cirugias, encontrar soluciones al problema de planificación de uso de estos recursos es vital para el funcionamiento de estas instituciones. Este problemas es conocido como Master Surgical Schedule (MMS) y es un problema NP-duro. Las soluciones propuestas son aproximadas usando diferentes heuristicas y son acompañadas con diferentes métricas para mostrar la eficiencia de estas.**

***Index Terms*—Master Surgical Schedule, Algoritmos, Heurísticas**

## I. Introduction

La administración hospitalaria debe enfrentarse a la tarea de asignar debidamente las salas de cirujía a los diferentes departamentos del hospital, este problema es de vital importancia para los hospitales modernos ya que estos son por lo general los recursos más limitados que estos poseen.

El objetivo de esta investigación es proveer a administradores de hospital con una herramienta básica de asignación que permita reducir los costos implicados en esta area, para este fin se crea una comparativa entre distintas soluciones que sean viables para un hospital de tamaño medio basándose en datos reales proveídos.

## II. Formulación del problema

El problema de Master Surgical Schedule (MMS) busca determinar una linea de tiempo para el uso de salas de cirugias basadas en un grupo de datos de pacientes sobre un el horizonte de planeación. Optimizar el tiempo de las salas de cirugias es entendido como maximar el número de casos de cirugia completados en el menor tiempo posible teniendo en cuenta la prioridad de estos.

Se busca que el algoritmo aplicado a MSS ajuste las expectativas de uso de salas de cirugías respecto a nuevos pacientes o cambios en los datos de urgencia y duración de los pacientes actuales. Los pacientes proporcionados superan el tiempo disponible de en el horizonte de planeación, por lo tanto la solución de MSS debe sugerir cuales pacientes son elegidos para ser atendidos en cada semana en un calendario cíclico.

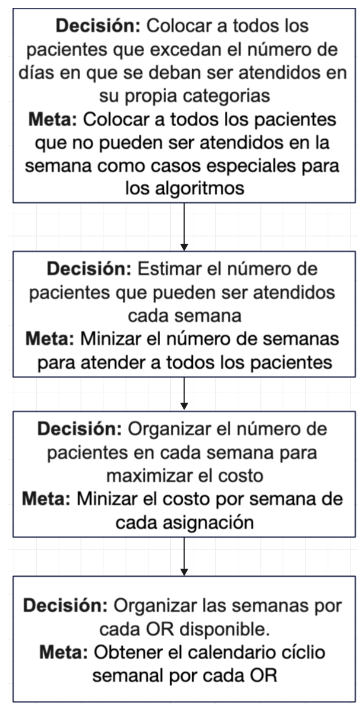



Fig 0. Diagramas de bloques de la formulación del problema

### II-A. *Descripción de datos*

El grupo de salas de cirugia $O = \{1, 2, \cdots, m\}$. El grupo de pacientes $P = \{p_1, \cdots, p_n\}$. Para cada $p \in I$ tiene las siguientes caracteristicas (1) $l$ Longitud de la estadia (LOS) donde $l \in \{1, 2, \cdots, 6\}$, (2) $u$ Prioridad del paciente con $u \in \mathbb{Z}$ y (3) Urgencia de atención $u \in \{D, C, B, A1, A2\}$.

### II-B. *Restricciones del problema*

El horizonte de planeación es cíclico, por lo tanto una tarea no se puede interrumpir mientras se encuentra en ejecución ejecuta y cada tarea puede ejecutarse hasta su finalización antes de pasar a la siguiente. [1] El horizonte de planeación es de 5 días con $n$ salas disponibles para todo el departamento.

### II-C. *Formato de la solución*

Se deben proporcionar los siguientes datos, (1) Cuantas y cuales salas de cirugias de las disponibles deben ser asignadas a cada día del hospital por cada sala, (2) Cuantos y Cuales son los pacientes que deben ser seleccionados. La solución del plan maestro es un arreglo $M \times N$ donde $M$ representa los dias planeados con $|M| = 5$ y $N$ representa las salas de cirugia con $|N| = n$

La solución debe optimizar el tiempo en que las salas estan ocupadas y a su vez tener en cuenta la prioridad de los pacientes, el peso que cada una de estas variables tiene dependerá de la implementación de los algoritmos.

## III. Formulación del Algoritmo heurístico

MSS puede ser formulado como un problema de búsqueda en el cual (1) Cada combinacion de asignaciónes de citas sobre el horizonte de planeación constituye un estado. (2) El estado incial siendo el horizonte completamente vacío sin ningún paciente asignado. (3) El modelo de transición toma el horizonte actual y añade una semana de asignacion de pacientes. (4) El estado meta es el calendario completado con todos los pacientes con citas. (5) El costo de cada acción es la cantidad de pacientes que no son operados en el horizonte actual.

### *III-A. Modelo para MSS*

El modelo formulado se basa en uno formulado por Tànfani, E. y Testi, A. [2], donde describe las asignaciones de pacientes en calendario en valores $0, 1$. En este modelo la relación de pacientes en el calendario esta dado por:

$$x_i = \begin{cases} 1 & \text{Si paciente i esta asignado} \\ 0 & \text{De otra forma} \end{cases}$$

Y si la función $0, 1$ del paciente si esta asignado al horizonte:

$$z_i = \begin{cases} 1 & \text{Si el paciente i no esta asignado en el horizonte} \\ 0 & \text{De otra forma} \end{cases}$$

De esta forma el algortimo busca minimizar el valor de $z$ para cada paso. La formula de costo sería entonces:

$$\text{Z} = \sum_{i}^{n} \frac{p_i}{l_i} z_i - \sum_{i}^{n} \frac{p_i}{l_i} x_i$$

Por lo tanto la implementación de cualquier algoritmo que busca resolver MSS desea minimizar el valor de $Z$ en cada paso. Para evaluar la efectividad de un algoritmo se tienen en cuenta el número de semanas $W$ que le toma al algoritmo atender a todos los pacientes.

### *III-B. Restricciones del problema*

1. Para todos los pacientes: $\forall i = \{1, 2, \cdots n\}$, se tiene que un paciente puede estar asignado en el algún tiempo del horizonte de planeación ó no esta por fuera.

$$\sum_{k=1}^{c} \sum_{t=1}^{b} x_{ikt} + z_i = 1$$

2. Para todos los LOS, $\forall h = 2, 3, 4, 5$ se tiene que ningún paciente puede ser asignado a una sala de cirugia en un dia $t$, si este no puede ser dado de alta antes del fin de semana. $I_h$ son todos los pacientes con LOS $h$ y $T_h$ son todos los días en que los pacientes con LOS $h$ no puede ser operados.

$$\sum_{i \in I_h} \sum_{k=1}^{c} \sum_{t \in T_h} x_{ikt} = 0$$

3. Ningún paciente puede estar sin asignación de día en el horizonte de planeación. por lo tanto para cualquier solución se tiene que:

$$\sum_{z \in Z} z_i = 0$$

## IV. Solución del algoritmo Heurístico Greedy

Sean los pacientes $P = \{p_1, \cdots, p_n\}$, cada paciente $p_i$ tiene un $LOS$ dado por $l_p$ y una urgencia dada por $u_p$. El algoritmo Greedy se compone de dos pasos:

### *IV-A. Paso 1:*

Se obtienen los pesos relativos dados de los pacientes dados por la ecuación:

$$c_p = u_p \ l_p$$

Luego se ordenan todos los pacientes en orden ascendente con estos pesos.

### *IV-B. Paso 2:*

Con el set de pacientes ordenados $P_o = \{p_1, p_2, \cdots, p_n\}$ se dividen en un set de soluciones posibles dados por:

$$H = \{\{p_1, \cdots, p_{k_1}\}, \{p_{k_1}, \cdots, p_{k_2}\} \cdots, \{p_{k_{n-1}}, \cdots, p_{k_n}\}\} https://wu$$

Donde para cada $h \in H$ contiene el máximo número de paciente $p$ tales que:

$$\sum_{p \in h} c_p \leq 5$$

Luego se divide $H$ en el número máximo de salas $m$ de operaciones tales que

$$M = \{(h_1, \cdots, h_m), (h_{m+1}, \cdots, h_{m+2}), \cdots (h_{k'}, \cdots h_{k'+m})\}$$

Por lo tanto se tiene que $M$ constituye una solución al problema MSS

## V. Solución del algoritmo Heurístico MTHM

El algoritmo de Martello y Toth(1980) [4] se construye basado en el algoritmo Greedy anterior y se basa en barajar los elementos en respuesta en de este. De esta forma se reducen las permutaciones del espacio respuesta, no tenemos que tener en cuenta todas las posibles combinaciones de pacientes sino la de algunos grupos de pacientes preseleccionador por el algoritmo Greedy.

El algoritmo MTHM se constituye de tres pasos:

### *V-A. Paso 1*

En la primera fase se preseleccionan los pacientes con el algoritmo greedy, de esta forma tenemos el grupo $H$ la cual se tomara como la solución aproximada.

### V-B. *Paso 2*

Para la nueva configuración de $M$ se busca maximizar el número de pacientes por cada horario, de esta forma para cada $h_i, h_j \in m$ con $j \neq i$ se tiene que

$$h_i = (p_i, \cdots, p_n), h_j = (p'_i, \cdots, p'_m)$$

Entonces

$$h'_i = (p_0, \cdots, p_n, p'i), h'_j = (p_0, \cdots, p_{m-1})$$

Si $h'_i$ y $h'_j$ cumplen con las restricciones del problema entonces estas reemplazan a $h_1, h_j$

### V-C. *Paso 3*

Para todos los elementos $h_i, h_j \in H$ con $j \neq i$

$$h_j = (p_1, \cdots, p_k, \cdots p_n), h_i = (p'_1, \cdots, p'_k, \cdots p'_m)$$

Se busca una nueva combinación de elementos dada por

$$h'_j = (p_1 \cdots, p'_k, \cdots, p_n), h'_i = (p_1 \cdots p_k \cdots, p_m)$$

Si $h'_i$ es una mejor solución que $h_i$ entonces se reeplaza por esta, y $h'j$ reemplaza a $h_j$. Para todos los elementos $p_k$ con $0 \leq k \leq min(m, n)$ se aplica este paso. Idealmente este swapping elemento a elemento resultará en una mejor solución.

### V-D. *Formato de respuesta*

Dados los nuevos elementos de $H$, se reducen las respuestas eliminando a todos los elementos vacíos. Luego se agrupan los elementos de $H$ en grupos de $m$ elementos para obtener el conjunto de respuesta a MSS

## VI. Resultados experimentales

En la evaluación comparativa ambos algoritmos posee los siguientes datos, el número de salas de operación son $|O| = 2$, con un número de pacientes $|P| = 80$, la urgencia de los atención esta dada por $u \in \{1, 2, \cdots\}$ con un horizonte de planeación de 5 días, cada paso es evaluado por el costo y el tiempo que logran reducir. Subsecuentes resultados varian el número de pacientes n y comparan los tiempos de estos.

Ambos algortimos fueron testeados en Ryzen 5900X con 4.8GHz y 32GB de memoria RAM. Los datos para la segunda prueba son generados usando una copula gaussiana sobre los datos proveídos por el paper de Tàfani, E y Tesi A. [2], los datos sintéticos son usados en lugar de datos reales debido a la dificultad que se tiene al adquirir estos de hospitales reales.

El tiempo de cada paso es tomado como el tiempo que le toma al algoritmo organizar la data y fraccionarla en un elemento del horizonte de planeación, de esta forma el tiempo del paso $i$ mide el tiempo que el algoritmo puede generar el output de la semana $i$.

### VI-A. *Heurística Greedy*

| PASO | COSTO | TIEMPO |
|---|---|---|
| 1 | 18236.71 | 0.0496 s |
| 2 | 10159.22 | 0.0498 s |
| 3 | 7915.22 | 0.05 s |
| 4 | 6581.72 | 0.0502 s |
| 5 | 4760.72 | 0.0503 s |
| 6 | 4055.72 | 0.0504 s |
| 7 | 3266.72 | 0.0506 s |
| 8 | 2806.72 | 0.0507 s |
| 9 | 2523.72 | 0.0509 s |
| 10 | 2025.72 | 0.051 s |
| 11 | 1800.72 | 0.0511 s |
| 12 | 1605.72 | 0.0513 s |
| 13 | 1343.72 | 0.0514 s |
| 14 | 1103.72 | 0.0516 s |
| 15 | 889.72 | 0.0517 s |
| 16 | 774.22 | 0.0519 s |
| 17 | 628.72 | 0.052 s |
| 18 | 496.72 | 0.0522 s |
| 19 | 375.92 | 0.0523 s |
| 20 | 206.92 | 0.0524 s |
| 21 | 116.92 | 0.0526 s |
| 22 | 57.58 | 0.0527 s |
| 23 | 34.25 | 0.0529 s |
| 24 | 18.0 | 0.053 s |
| 25 | 8.0 | 0.0532 s |
| 26 | 2.5 | 0.0533 s |

Fig 1. Resultados de pasos Greedy en caso de 80 pacientes

| PACIENTES | SEMANAS | TIEMPO |
|---|---|---|
| 80 | 26 | 0.0533 s |
| 200 | 74 | 0.0498 s |
| 500 | 173 | 1.2280 s |
| 1000 | 349 | 2.5601 s |
| 5000 | 1759 | 12.1593 s |
| 10000 | 3553 | 24.2893 s |

Fig 2. Resultados de pasos Greedy en caso de número variante de pacientes

### VI-B. Heurística MTHM

| PASO | COSTO | TIEMPO |
|---|---|---|
| - 1 | 18236.72 | 0.3382 s |
| 2 | 9619.22 | 0.3481 s |
| 3 | 7442.72 | 0.3563 s |
| 4 | 5942.72 | 0.3637 s |
| 5 | 4100.72 | 0.3718 s |
| 6 | 3266.72 | 0.377 s |
| 7 | 2428.72 | 0.3857 s |
| 8 | 2061.72 | 0.3937 s |
| 9 | 1689.72 | 0.4025 s |
| 10 | 1391.72 | 0.4088 s |
| 11 | 1077.72 | 0.4179 s |
| 12 | 868.72 | 0.4236 s |
| 13 | 680.72 | 0.4282 s |
| 14 | 515.72 | 0.4356 s |
| 15 | 377.72 | 0.4421 s |
| 16 | 267.72 | 0.4489 s |
| 17 | 156.92 | 0.4553 s |
| 18 | 72.25 | 0.4612 s |
| 19 | 45.58 | 0.4678 s |
| 20 | 24.25 | 0.4728 s |
| 21 | 12.0 | 0.4762 s |
| 22 | 5.0 | 0.4808 s |
| 23 | 2.5 | 0.5067 s |

Fig 3. Resultados de pasos Greedy en caso de 80 pacientes

| PACIENTES | SEMANAS | TIEMPO |
|---|---|---|
| 80 | 23 | 0.3 s |
| 200 | 63 | 1.3 s |
| 500 | 162 | 6.7 s |
| 1000 | 316 | 23.24 s |
| 5000 | 1550 | 357.72 s |
| 10000 | 3066 | 1444.09 s |

Fig 4. Resultados de tiempo y eficiencia(Semanas) del algoritmo MTHM

### VI-C. Tabla Comparación de tiempos y eficiencia

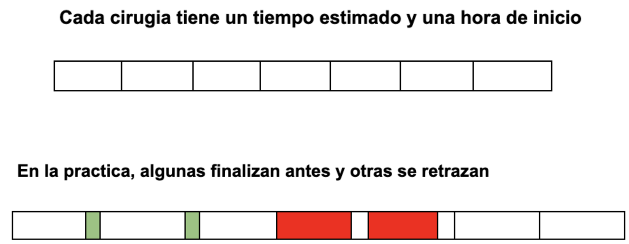
| PACIENTES | % EFFICIENCIA | % TIEMPO |
|---|---|---|
| 80 | 11.53 | 562.85 |
| 200 | 14.86 | 2610.44 |
| 500 | 6.35 | 545.60 |
| 1000 | 9.45 | 907.77 |
| 5000 | 11.88 | 2941.94 |
| 10000 | 13.70 | 5945.37 |

Fig 5. Comparativa de tiempo y eficiencia entre algoritmos Greedy y MTHM

## VII. Formulación del Algoritmo Genetico

### VII-A. Introducción al problema

Se realizan cirugías que tienen tiempos de ejecución estimada entre 1 y 5 horas, siempre horas enteras y una hora máxima para su realización que va entre 5 y 24*5, que son las horas correspondientes a cinco días. Ambos tiempos corresponden a una distribución uniforme. Durante la ejecución a veces hay sobretiempos o sub tiempos, que obedecen a una distribución normal. El objetivo es minimizar Z=sobretiempo total para la sala.




### VII-B. Implementación

Se generan 24*5*6/3 cirugias (6 quirofanos, 5 dias, 3 horas de duración la cirugía promedio) cada una con un limite de finalización (distribución uniforme) entre 5 y 24*5. Inicialmente se ordenan por prioridad (Con un Bubble sort) , y se asignan en orden a cada quirófano, finalmente se crea una asignación inicial a cada quirófano

### VII-C. Simulación

Tenemos inicialmente las cirugías asignadas a cada quirófano, algunas tienen sobretiempo y otras tienen un tiempo de sala vacia. A cada cirugía se le asigna con una distribución normal un retrazo/adelanto. Y se mide el sobretiempo total. Nótese que los pacientes llegan a la hora asignada, es decir si la sala por un adelanto queda vacia, no es utilizada.

### VII-D. Genético

Realizamos una mutación sobre las asignaciones (Tomando al azar una cirugía de los quirófanos con más sobretiempo y pasandolas a aquellas asignaciones que generan menos sobretiempo en la simulación) Posteriormente mezclamos todas las asignaciones, para un total de (5+4+3+2+1) Hijos. Simulamos de nuevo y sobreviven solo las 6 mejores asignaciones, repetimos el proceso.

La mezcla de las asignaciones se hace así:

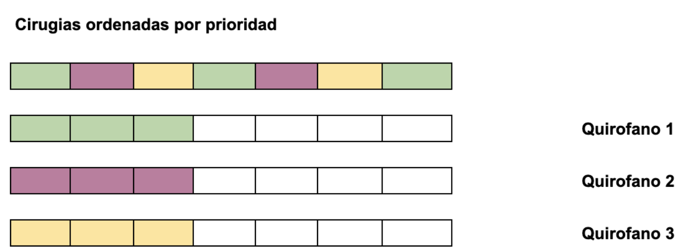



Fig 6. Mezcla de asignación de calendario

### VII-E. Resultados

Inicialmente los tiempos de retrazo promedio total en promedio para los seis quirofanos son de 80 horas, después de implementar este algoritmo genetico con 100 generaciones tenemos tiempos de demora estimados en 1 hora.

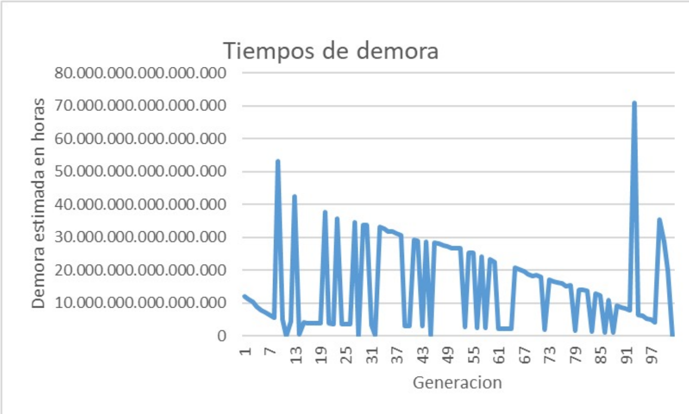



Fig 7. Resultados de los tiempos de demora del algoritmo genético

## VIII. Conclusiones

Los resultados de ambos experimentos muestran que mientras la Heurística MTHM obtiene una solución más adecuada, en promedio un $10\,\%$ más eficiente que la solución Greedy, sin embargo el tiempo de computación de esta incrementa drásticamenta. El tiempo de computo de la solución MTHM es muy dependiente de que tan eficiente sea la primera solución dada por el algorimto Greedy, de esta forma se explica la anomalia en los datos de $n = 500$. Por lo tanto dependiendo de la cantidad de pacientes resulta beneficioso usar el algoritmo $MTHM$ sobre la alternativas Greedy, sin embargo la cantidad de pacientes pueden hacer que esta solución sea inviable en su forma actual. El algoritmo $MTHM$ puede limitar su búsqueda de alternativas en el paso 2 y 3 para evitar que incremente drásitcamente el campo de búsqueda de soluciones alternativas, este puede ser un punto intermedio viable.

La solución Greedy al problema MSS parece adecuada, esto se debe a que la distribución actual de los datos proveídos que hacen que la organización de pesos relativos se aproximen bastante a la solución. En otras distribuciones de datos menos favorables la solución Greedy puede ser bastante menos viable.

## Referencias


[1] Wanhammar, L. (1999). Chapter 7: DPS Design System. In DSP integrated circuits. essay, Academic Press.

[2] Tànfani, E., y Testi, A. (2009). A pre-assignment heuristic algorithm for the Master Surgical Schedule Problem (MSSP). Annals of Operations Research, 178(1), 105–119. https://doi.org/10.1007/s10479-009-0568-6

[3] J. Csirik, J.B.G. Frenk, M. Labbe, S Zhang (1991). Heuristics for the 0-1 Min-Knapsack Problem. Acta Cybernetica, Tom. 10, Fasc. 1-2, Szeged, 1991

[4] Lalami, M. E., Elkihel, M., Baz, D. E., y Boyer, V. (2012). A procedure-based heuristic for 0-1 multiple Knapsack Problems. International Journal of Mathematics in Operational Research, 4(3), 214. https://doi.org/10.1504/ijmor.2012.046684