MINIMIZACIÓN DE LA TARDANZA EN PROBLEMAS DE PROGRAMACIÓN DE TAREAS EN MAQUINAS PARALELAS CON DETERIORO DE LOS RECURSOS

MINIMIZING TARDINESS IN PARALLEL MACHINE SCHEDULING WITH RESOURCE DETERIORATION

Alex J. Ruiz-Torres[1,♦], José H. Ablanedo-Rosas[2], Nelson Alomoto[3], Diana Jadan Avilés[4]

RESUMEN

En ambientes de manufactura y de servicios es frecuente encontrar diferentes tareas que son realizadas en paralelo empleando recursos heterogéneos, los cuales tienen la característica de sufrir deterioro a medida que transcurre el tiempo. Ese deterioro tiene un impacto significativo en el desempeño de dichos recursos, lo que se puede medir de diferentes formas tales, como: calidad, tiempo de proceso, entre otros. Esta investigación científica utiliza un modelo donde el deterioro de los recursos es una función de las tareas específicas previamente realizadas por el recurso. La formulación del problema se presenta por medio de un modelo de programación matemática. Este trabajo presenta dos heurísticas para resolver el problema en un tiempo razonable, donde cada heurística emplea diferentes reglas y criterios para identificar la mejor solución. Un análisis de sensibilidad, que comprende 2700 casos, es llevado a cabo para evaluar la eficacia de las heurísticas. Los resultados comprueban que las heurísticas son eficientes y generan soluciones útiles para el tomador de decisiones.

Palabras Claves: Deterioro de máquinas, fechas de entrega, tardanza, máquinas paralelas, programación de operaciones, heurísticas.

ABSTRACT

In manufacturing and service environments it is common to find processes that are performed in parallel by different resources, which have the characteristic that their performance deteriorates with time. This deterioration has a significant effect on the performance of the resources that can be measured in different forms such as quality and process time. This research utilizes a model where resource deterioration is a function of the specific jobs previously completed by the resource. The problem's formulation is presented as a mathematical program. The paper presents two heuristics to solve the problem, where each has different rules to find the best solution. A sensitivity analysis that includes 2700 cases is performed to evaluate the performance

[1]Facultad de Administración de Empresas. Universidad de Puerto Rico – Rio Piedras. San Juan, PR 00931-3332, USA. alex.ruiztorres@upr.edu

[2]Department of Marketing and Management.College of Business Administration.University of Texas at El Paso.TX 79968, USA. jablanedorosas2@utep.edu

[3]Facultad de Ciencias Administrativas. Escuela Politécnica Nacional. Quito, 17-01-2759 Ecuador. nelson.alomoto@epn.edu.ec

[4]Carrera de Ingeniería Industrial. Facultad de Ciencias Químicas. Universidad de Cuenca. Ecuador. diana.jadan@ucuenca.edu.ec

♦Autor para correspondencia:alex.ruiztorres@upr.edu

Recibido: 26.08.2013 Aceptado: 28.02.2014

of the heuristics. The results demonstrate that the heuristics are efficient and generate useful solutions for decision makers.

Keywords: Resource deterioration, due dates, tardiness, parallel machines, scheduling, heuristics.

INTRODUCCIÓN

Una característica importante de los sistemas de producción es el deterioro que sufren los recursos utilizados, lo cual resulta en un desgaste que se transforma en una inferior capacidad de desempeño a medida que se utiliza el recurso o transcurre el tiempo. En algunos sistemas, este deterioro requiere de mucho tiempo para ser significativo, por lo cual no es un elemento importante en la programación de las tareas (programación de producción). Sin embargo, en otros sistemas este deterioro ocurre de manera rápida y su efecto crea complejidad y costos adicionales, si no es manejado de manera apropiada. Por ejemplo, en el caso de los seres humanos, el cansancio se acumula a medida que transcurre el día, requiriendo mayores niveles de esfuerzo y tiempo para realizar una misma labor. No sólo el deterioro puede resultar en un mayor tiempo de proceso, sino que también provoca un aumento de la probabilidad de error. En el caso de máquinas, los procesos mecánicos y de corte desgastan el equipo; por lo tanto, considerar el deterioro es muy importante.Los componentes de las máquinas se consumen o se vuelven más “lentos”, lo que resulta en problemas de calidad, ya que se requieren ajustes y/o reparaciones en el equipo.

Otra característica común en los sistemas de producción, sea en servicios o en manufactura, es la estrategia de procesamiento en paralelo. Por ejemplo, en el caso de servicios, los cajeros en un banco son recursos paralelos que procesan transacciones de clientes, donde cada cliente requiere un “trabajo” con un contenido diferente, pero donde cada cajero tiene las capacidades para atender a cualquiera de los clientes en la fila. Similarmente, en procesos de manufactura se pueden observar máquinas, por ejemplo de inyección de plástico, cada una procesando “órdenes” diferentes, pero donde cada una de esas máquinas (asumiendo un tonelaje idéntico) tiene la capacidad de procesar cualquiera de estas órdenes (lo diferente en cada máquina es el molde instalado).

Esta investigación se origina de observaciones de campo en diferentes ambientes de manufactura combinadas con una revisión de la literatura, donde se establece el problema de programación de tareas en recursos paralelos cuando estos recursos de producción se deterioran con el tiempo. Estudios que consideran el deterioro en los recursos de producción son abundantes en la literatura científica; los primeros trabajos fueron reportados por Gupta & Gupta (1988) y por Browne & Yechiali (1990); un par de artículos enfocados a la revisión de la literatura sobre el tema han sido publicados por Alidaee & Womer (1999) y Cheng *et al*. (2004). Los estudios previos revelan que el problema del deterioro en los recursos es relevante, ya que aumenta considerablemente el tiempo requerido para completar una tarea y, por consiguiente, se reduce la eficiencia operacional y la capacidad de satisfacer la especificaciones requeridas por el cliente.

El servicio al cliente, modelado por el tiempo promedio de la tardanza, es el objetivo de negocio que se investiga en este artículo. Por lo tanto, cada tarea a ser procesada tiene un tiempo de entrega comprometido, y el objetivo es minimizar la suma de los retrasos en los tiempos de entrega. Investigaciones relacionadas a la minimización de la tardanza total y otros objetivos relacionados han recibido la atención de muchos investigadores; revisiones de la literatura sobre problemas de programación de tareas, considerando fechas de entrega, han sido completadas por Biskup y Herrmann (2008), Sterna (2011) y Xu *et al.* (2010). El problema es muy relevante,

ya que es un caso frecuente que se observa en ambientes de manufactura y de servicios, y que tiene como prioridad el cumplimiento de los tiempo de entrega comprometidos con el cliente.

Existen numerosas investigaciones que estudian la programación de tareas y consideran el deterioro de los recursos de producción. Una suposición común en estos artículos publicados es que el deterioro se modela como una función del tiempo o del número de tareas que han sido procesadas por el recurso. Por ejemplo, a las 2 pm, el recurso *X* va a funcionar a un 90% de su nivel óptimo,independientemente de las tareas que haya completado desde las 8am; o el recurso *S* va a funcionar a un 76% de su nivel óptimo luego de completar 5 solicitudes de empleo, sin importar el contenido de éstas. Nuestro modelo de deterioro de recursos de producción está basado en el propuesto por Ruiz-Torres *et al.*(2013), donde el deterioro de los recursos es una función de las tareas específicas realizadas por el recurso.Si por la mañana al recurso *S* sólo le fueron asignadas tareas"fáciles", es posible que a las 2 pm esté funcionando a un 95% de su nivel óptimo, mientras que, si por la mañana el recurso *S* invirtió su tiempo en la solución de problemas complicados y discusiones acaloradas, es posible que a las 2 pm esté a un 70% de su nivel óptimo. Similarmente, en el caso de manufactura, si una celda se dedicó a ensamblar un producto complicado/nuevo, es posible que la celda se haya desgastado más que si se hubiera asignado un trabajo más sencillo o muy conocido.

Este artículo continúa y mejora la investigación de Ruiz-Torres *et al*. (2013) al estudiar el problema de programación de tareas en máquinas paralelas, considerando el deterioro de los recursos y teniendo como objetivo el minimizar la tardanza total. El objetivo en el estudio de Ruiz-Torres *et al.* (2013) es diferente y está enfocado a la minimización del tiempo total en el que se terminan todas las tareas (conocido como "makespan"). El "makespan" es una medida relacionada al uso eficiente de los recursos de producción y no toma en consideración las fechas de entrega; sólo se utilizan los tiempos de proceso de las tareas en cada máquina. Al no considerar las fechas de entrega, y por consiguiente la tardanza, coma parte de la función objetivo, una programación particular de tareas puede ser óptima para el "makespan", y a la vez ser ineficiente con respecto al nivel de servicio al cliente al presentar una tardanza total muy alta. Se entiende que en ambientes competitivos de manufactura y servicios es estratégico analizar problemas con deterioro de recursos y con una función objetivo enfocada en el servicio al cliente.

Los investigadores han utilizado diversos enfoques para abordar el efecto de deterioración de los recursos de producción. Unos de estos enfoques es caracterizar el tiempo de proceso como una función del tiempo de inicio de la tarea. Definimos p_j, p'_j, e_j, y $b_j$como el tiempo de procesamiento básico, el tiempo de proceso efectivo, el factor de deterioro, y el tiempo de inicio de la tarea *j*,respectivamente. Basado en estas definiciones, el tiempo de proceso efectivo ha sido modelado como una función lineal del tiempo de inicio; $p'_j = p_j + e_j b_j$ o $p'_j = e_j b_j$. Algunos de los estudios recientes que consideran el tiempo de proceso como una función lineal son: Toksari & Güner (2010), Mazdeh *et al.* (2010), Huang & Wang (2011), Joo & Kim (2012), Liu *et al*. (2013), Cheng *et al*. (2014) y Wu *et al*. (2013a).Toksari & Güner (2010) estudiaron el caso de máquinas paralelas con el objetivo de minimizar una fecha de entrega común, y considerando aprendizaje (tasa de aumento) y deterioro (tasa de decremento) simultáneamente; el factor de aumento o decremento es idéntico para todas las tareas (así e_j = *e*para todas las tareas). Mazdeh*et al*. (2010) analizaron el deterioro de trabajos en forma lineal, además de incluir el costo del deterioro de las máquinas; la función objetivo es la minimización de ambos, la tardanza total y el costo de deterioro de las máquinas. Joo & Kim (2013) investigaron el caso de deterioro lineal en máquinas paralelas donde existen actividades que modifican el desgaste (i.e. actividad de mantenimiento); la función objetivo es minimizar el "makespan". Liu *et al.* (2013) desarrollaron dos heurísticas para el problema de minimizar el "makespan" en el caso de máquinas paralelas, e investigaron diferentes relaciones entre grados de deterioro y el número de máquinas; estos autores reportan un límite fijo para el caso en que el número

de máquinas es menor que cinco. Cheng *et al.* (2014) estudiaron el sistema conocido como "flow shop", donde existen dos máquinas con el objetivo de minimizar la suma de los tiempos de terminación de las tareas, sujeto al mínimo makespan; este estudio propuso un modelo matemático y unos algoritmos para resolver el problema. Wu e*t al*. (2013b) analizaron el caso de una sola máquina, pero donde los trabajos tienen dos agentes, y el tiempo de proceso en cada agente se deteriora independientemente con el tiempo; el objetivo es la minimización del número de trabajos finalizados tarde debido al primer agente en combinación con una tardanza debida al segundo agente, la cual tiene un límite superior.

Otros trabajos de investigación han caracterizado el tiempo del proceso como una función de la posición de la tarea en la secuencia de la máquina. Definamos p'_{jrh} como el tiempo de proceso de la tarea *j*que es procesada en la posición *r* de la máquina *h*;en esta caracterización, el tiempo de proceso de una tarea se define por $p'_{jrh} = p_{jh} + r \times e_{jh}$, o por $p'_{jrh} = p_{jh} \times r^{ejh}$,dónde e_{jh} es el efecto de deterioro de la tarea*j*en la máquina *h*, y la posición *r* depende del número de tareas después de un evento de mantenimiento. Dentro de las investigaciones recientes que consideran modelos de deterioro basados en la posición de la tarea se encuentran:Yang (2011), Mosheiov (2012),Yang *et al.* (2012) y Yang (2013). Yang (2011) investigó la minimización de las sumas de los tiempos de terminación de tareas en cada máquina, tomando en cuenta la frecuencia de mantenimiento, donde *r* depende del plan de mantenimiento de cada máquina. Mosheiov (2012) estudió el mismo objetivo, pero el deterioro es determinado por una matriz basada en la posición de las tareas. Yang *et al.*(2012) también investigaron la suma de los tiempos de terminación de tareas en todas las máquinas, pero en este caso modelaron máquinas no relacionadas (no idénticas). Yang *et al.*(2013) expandieron los modelos anteriores al considerar la programación conjunta de las tareas y del mantenimiento para el caso de máquinas paralelas; el tiempo de mantenimiento de cada máquina depende del tiempo que ésta haya estado procesando tareas, la función objetivo es minimizar la suma del tiempo de finalización de todas las tareas.

La literatura científica que estudia la programación de tareas con criterios relacionados a fechas de entrega es bastante extensa, por lo cual nos limitamos a revisar la literatura que analiza la suma de las tardanzas en máquinas paralelas y está enfocada a problemas asociados a fechas de vencimiento,además de incluir la condición de deterioro de recursos. Artículos que estudian problemas con una sola máquina son abordados por varios investigadores. Eren & Güner (2007), incorporaron el efecto de aprendizaje a este tipo de problema y plantearon un modelo de programación entera para menos de 25 trabajos, mientras que para problemas grandes propusieron tres heurísticas basadas en búsqueda aleatoria, búsqueda tabú (TS) y recocido simulado (SA), respectivamente. Cheng *et al.* (2011), a través de un algoritmo de ramificación y acotamiento (Branch & Boundo BAB) resolvieron el problema para trabajos con deterioro y tiempos de preparación lineales, con la particularidad de que el objetivo es minimizar la tardanza máxima. Wu *et al*. (2013b) investigaron el problema de una sola máquina con deterioro lineal y tiempos distintos de emisión de las tareas, desarrollaron un algoritmo de ramificación y acotamiento (BAB) y propusieron una hibridación con un algoritmo de colonia de abejas para encontrar una solución cercana al óptimo.

Trabajos que abordan problemas de la suma de las tardanzas en máquinas paralelas muestran diversos enfoques. Los estudios de Azizoglu & Kirkca (1998), Yalaoui & Chu (2002), y Shim & Kim (2007) utilizaron algoritmos de ramificación y acotamiento (BAB); el primero presenta propiedades que caracterizan la solución óptima, el segundo expone esquemas de acotamiento superior e inferior para la solución, y el tercero utiliza cotas superior e inferior obtenidas heurísticamente. Para el mismo tipo de problema,Tanaka & Araki (2008) desarrollaron un algoritmo BABy para determinar una cota inferior utilizaron el método de relajación de Lagrange; este algoritmo se utilizó para resolver problemas con alrededor de 25 tareas y cualquier número de máquinas. Biskup *et al.* (2008) propusieron una nueva heurística, la cual se sometió a

prueba con tres algoritmos conocidos: TPI (índice de prioridad de tráfico), MDD (fecha de vencimiento modificada) y KPM (descomposición y heurística híbrida);los resultados mostraron un desempeño superior de la heurística propuesta; de igual forma, con los datos de prueba se determinó un mejor desempeño de MDD sobre TPI y KPM al minimizar la tardanza total para problemas de máquinas paralelas idénticas. El caso de máquinas paralelas idénticas, con fechas de vencimiento común y pesos o penalidades proporcionales, fue analizado por Sun & Wang (2003). En este trabajo, la minimización de la suma ponderada de tiempos tempranos y tardanzas se examinó a través de un algoritmo de programación dinámica; además, se desarrollaron dos heurísticas basadas en listas de programación (LS) para enfrentar la problemática y analizar las cotas de error para el peorescenario. En un problema relacionado, Vélez & López (2011) presentaron un algoritmo para programar un conjunto de *n* tareas en una máquina de procesamiento por lotes con capacidad específica, de tal manera que la tardanza total ponderada sea mínima. El método utilizado es un heurístico de búsqueda de entorno variable descendiente (BEVC), que parte de una solución inicial para luego mejorarla por medio de la exploración sistemática de múltiples vecindarios. Para mejorar el desempeño del algoritmo se propuso ejecutarlo múltiples veces, cada una con una solución inicial diferente, y almacenar la mejor solución encontrada por el algoritmo durante las múltiples ejecuciones.Della Croce *et al.* (2012), abordaron la minimización de la tardanza total ponderada, desarrollando un algoritmo que se basa en el intercambio generalizado de pares, optimización de búsqueda dinámica y vecindades de máquinas paralelas. El problema con tiempos diferentes de emisión de las tareas fue investigado por Jouglet & Savourey (2011); ellos describieron reglas de dominancia y métodos de filtrado; además, mostraron como deducir si una tarea puede ser procesada en una máquina determinada.

Los problemas de máquinas paralelas no-relacionadas son un caso más complicado que el problema de máquinas paralelas idénticas (Zhang *et al*. 2012). Bank y Werner (2001)estudiaron el caso donde los tiempos de proceso dependen de la máquina que los procesa; las fechas de inicio están definidas para cada tarea, y se asume una fecha de vencimiento común. El objetivo es minimizar la suma ponderada de los tiempos tempranos lineales y las tardanzas penalizadas. En el estudio se probaron varios algoritmos para casos de hasta 50 tareas y 200 máquinas, y los resultados no revelaron ninguna superioridad de un algoritmo sobre otro.Liaw *et al*. (2003), investigaron el problema de máquinas paralelas no-relacionadas, que tiene dificultad-NP en sentido fuerte, y plantearon dos cotas, una inferior resultante de un problema de asignación, y una superior obtenida con una heurística de dos fases; además, probaron un algoritmo BAB que usa reglas de dominancia. Se obtuvieron buenos resultados para problemas con un máximo de 18 tareas y 4 máquinas.Bilge *et al.* (2004) analizaron una versión generalizada de máquinas paralelas, donde las tareas tienen fechas de vencimiento y tiempos de arribo distintos, y presentaron un algoritmo TS que se probó con algunos problemas disponibles en la literatura. El algoritmo propuesto presentó un desempeño superior a los algoritmos reportados hasta ese momento. De igual forma, Anghinolfi & Paolucci (2007) estudiaron el problema con tiempos de preparación diferentes de cero y configuraciones dependientes de una secuencia. Para este caso plantearon un enfoque meta-heurístico híbrido, que incluye los siguientes algoritmos: TS, SA y búsqueda local variable. Los experimentos muestrearon que el algoritmo propuesto tiene un desempeño aceptable frente al problema generalizado.

El Problema

En esta sección se describe el problema de forma detallada. El modelo de programación matemática que se presenta tiene como base el desarrollado por Ruiz-Torres *et al.* (2013), en el cual existen *n* tareas a ser asignadas a *m* recursos en paralelo.Se definen *N* y *M* como los conjuntos de tareas y recursos (máquinas), respectivamente; $N = \{1, \ldots, n\}$ y $M = \{1 \ldots m\}$. Las tareas no pueden ser divididas entre los recursos en paralelo, es decir, cada tareadebe ser asignada a un solo recurso. Además, una vez que comienza la tarea en el recurso, no

se puede detener su procesamiento. Se asume que todas las tareas están disponibles en el tiempo 0. Cada recurso tiene la capacidad de procesar una sola tarea a la vez. Estos recursos se consideran no-idénticos, por lo que el tiempo base de procesamiento de la tarea depende del recurso en donde se procesa la tarea; el tiempo de procesamiento base de la tarea *j* en el recurso *k* es p_{jk}.

A diferencia del problema estudiado por Ruiz-Torres *et al.* (2013), este trabajo asume que cada tarea tiene su fecha de entrega; se define d_j como la fecha de entrega de la tarea *j*. Se define e_{jk} *como* el efecto del deterioro de la tarea *j* en el recurso *k*. Se asume que e_{jk} puede asumir un valor de 0 o mayor, pero menor que 1. Un efecto de deterioro de 0 (ie. e_{jk} = 0) indica que la tarea *j* no deteriora el recurso *k*, en otras palabras, seguirá funcionando al mismo nivel que cuando se comenzó la tarea *j*. Un valor de e_{jk} > 0 indica el porcentaje de desgaste sobre el nivelde la "pre-tarea actual" que el recurso tendrá luego de completar la tarea *j*. Por ejemplo, un valor de e_{jk} = 0,1 indica que el recurso *k* se desgastará un 10% al terminar la tarea *j*.

Se asume que $n>m$ y que por lo menos se le asigna una tarea a cada recurso, por lo cual la asignación máxima a cualquiera de los recursos es de $n - m + 1$ tareas. Se define $g = n - m + 1$ como el número de posibles posiciones en la programación de cualquier recurso y sea $G = \{1, \ldots, g\}$. Se especifica x_{jkh} como una variable binaria que define si la tarea *j* fue asignada al recurso *k* en la posición *h,* y es una condición básica que $\sum_{h\in G,\, k\in M} x_{jkh} = 1 \forall j \in N$, de forma que que cada tarea es asignada sólo a un recurso y a una sola posición en la programación.

Se identifica la variable q_{kh} como el nivel de rendimiento del recurso *k* para la tarea en la posición *h*. Al comienzo del plan de trabajo, cada recurso comienza a su nivel "óptimo". En otras palabras, con ningún deterioro; por consiguiente, q_{k1} = 1 para todos los recursos *k* en *M*. El rendimiento del recurso *k* en la posición *h* (*h*> 1) se determina por la multiplicación del rendimiento en la posición anterior por el deterioro causado por la tarea en la posición anterior: $q_{kh} = \sum_{j\in N} (1 - e_{jk}) \times x_{jk(h-1)} \times q_{k(h-1)}$. El tiempo de proceso actual de la tarea *j* asignada a la posición *h* en recurso *k* es igual a p_{jk}/q_{kh}.

La función objetivo analizada en el trabajo de Ruiz-Torres *et al.* (2013) es la minimización del makespan, el cual es un objetivo relacionado con la eficiencia y utilización de los recursos. El enfoque de la investigación actual es la programación de tareas que maximizan el servicio al cliente, el cual puede ser modelado con la minimización del promedio de todas las tardanzas. Se define c_j como el período de tiempo para que la tarea *j* sea terminada. La tardanza de una tarea es t_j, $t_j = max.\ [c_j - d_j, 0]$, es decir, un valor positivo si la tarea es terminada después de la fecha de entrega, y 0 si la tarea está a tiempo.Se define la tardanza promedio como $t_{ave} = (1/n) \times \sum_{j\in N} t_j$. En la formulación matemática que se presenta a continuación no se definen los índices por tarea (ie. índice *j*), sino por el equivalente, que es la tarea asignada en la posición *h* en el recurso *k*.

$$\text{Minimizar } t_{ave} = (1/n) \times \sum_{h\in G,\, k\in M} t_{hk} \qquad (1)$$

$$\sum_{j\in N} x_{jkh} \le 1 \qquad \forall h\in G,\ k\in M \qquad (2)$$

$$\sum_{h\in G,\, k\in M} x_{jkh} = 1 \qquad \forall j\in N \qquad (3)$$

$$x_{jkh} \le \sum_{j\in N} x_{jk(h-1)} \qquad \forall j\in N, k\in M, h\in G\backslash\{1\} \qquad (4)$$

$$q_{kh} = \sum_{j\in N} (1 - e_{jk}) \times q_{k(h-1)} \times x_{jk(h-1)} \qquad \forall h\in G\backslash\{1\}, k\in M \qquad (5)$$

$$q_{k1} = 1 \qquad \forall k\in M \qquad (6)$$

$$t_{kh} \ge \sum_{j\in N,\, l = 1..h} p_{jk}/q_{kl} \times x_{jkl} - \sum_{j\in N} d_j \times x_{jkh} \qquad \forall k\in M, h\in G \qquad (7)$$

$$t_{kh} \ge 0 \qquad \forall k\in M, h\in G \qquad (8)$$

$$q_{kh} \ge 0 \qquad \forall k\in M, h\in G \qquad (9)$$

$$x_{jkh} \in \{0, 1\} \qquad \forall j\in N, k\in M, h\in G \qquad (10)$$

En el modelo matemático, la ecuación (1) es la función objetivo. La desigualdad (2) indica que,a cada posición en cada recurso puede asignársele máximo una tarea, mientras que la ecuación (3) indica que cada tarea debe asignarse a una posición en un recurso. La desigualdad (4) garantiza las asignaciones continuas. Las ecuaciones (5-6) definen el nivel de rendimiento de cada recurso para cada posición de la tarea. Las desigualdades (7-8) establecen el valor de la tardanza de las tareas. En la desigualdad (7), la componente $\sum_{j\in N,\ l=1..h} p_{jk}/q_{kl} \times x_{jkl}$ representa el tiempo en que se termina la tarea asignada a la posición/recurso *kh*, y cuando ninguna tarea está asignada a esa posición, el valor es 0. La componente $\sum_{j\in N} d_j \times x_{jkh}$ establece la fecha de entrega, y de igual manera si ninguna tarea está asignada a la posición/recurso *kh*, el valor es 0. La desigualdad (9) establece que el nivel de rendimiento no puede tener un valor negativo. La ecuación (10) define las variables binarias. El problema tiene $nm(n\text{-}m+1)$ variables binarias, haciéndolo ”computacionalmente difícil”cuando *n* y *m* aumentan.

Métodos de Solución

Esta sección presenta los métodos de solución para el problema presentado en la sección anterior. Como base de estos métodos se consideran varias heurísticas que se describen en el artículo de Biskup *et al.* (2008). Nuestra investigación considera sistemas donde los tiempos de proceso de cada tarea pueden ser diferentes para cada recurso, mientras que los trabajos anteriores consideran tiempos de proceso idénticos para cada recurso; por tanto, las heurísticas propuestas anteriormente no pueden ser aplicadas directamente en la solución del problema propuesto.

Como método de solución proponemos dos estrategias “básicas”: asignar las tareas considerando simultáneamente todos los recursos y asignar las tareas a los recursos uno por uno. Igual que en heurísticas presentadas en trabajos anteriores que consideran el promedio de las tardanzas y el deterioro de los recursos, se deben examinar todas las posibles posiciones dentro de una secuencia cuando se añade una tarea a una secuencia existente. Además, se utilizan estrategias similares en la selección de tareas a heurísticas propuestas anteriormente.

Reglas de prioridad

Un sinnúmero de heurísticas de programación de tareas se basa en la asignación de tareas a los recursos basándose en reglas de prioridad, las cuales están relacionadas a características de las tareas y/o de los recursos. De las reglas de prioridad más conocidas estála SPT (por sus siglas en inglés para “shor test processing time”) y la EDD (por sus siglas en inglés para “earliestdue date”). La regla SPT simplemente le da la prioridad a la tarea con el tiempo de proceso más corto, mientras que la regla EDD le da prioridad a la tarea con fecha de entrega más próxima. En el caso de la minimización del promedio de las tardanzas, una regla frecuentemente utilizada es la TPI (por sus siglas en inglés para “traffic priorit yindex”) que caracteriza la congestión del sistema para darle peso al tiempo de proceso y a la fecha de entrega, creando un índice de prioridad para cada tarea, basándose en esta información. En adición a estas tres reglas de prioridad, en esta investigación se considera el parámetro de deterioro de las tareas (e_{jk}), donde se le da prioridad a las tareas con menor deterioro.

Dado que nuestra investigación considera recursos no relacionados (o no idénticos), se determina un TPI particular para cada recurso y uno general, como se describe a continuación. Se define el tiempo mínimo de proceso de una tarea como $p^*_j = min._{k\in M}[p_{jk}]$. El total de los tiempos de proceso en un recurso es $P_k = \sum_{j\in N} p_{jk}$ y el total de los tiempos mínimos es $P^* = \sum_{j\in N} p^*_j$. Se define el total de los tiempos de entrega como $D = \sum_{j\in N} d_j$. Para cada recurso *k* se determina el nivel de congestión, $NC_k = n \times P_k / (m \times D)$ y para el sistema en general $NC^* = n \times P^* / (m \times D)$ $\forall k \in M$. Utilizando la constante 3 como en Ho and Chang (1991) and Biskup *et al.* (2008), el

peso de las fechas de entrega para el recurso k es $WD_k = max$. [0, min [0.5 +(3 – NC_k)/ NC_k, 1]] y el peso para los tiempos de proceso es$WP_k = 1 – WD_k$. De igual manera, el peso de las fechas de entrega para el sistema en general es $WD^* = max$. [0, min [(3 – NC^*) / NC^*], 1]] y el peso para los tiempos de proceso es $WP^* = 1 – WD^*$.

Se define el tiempo máximo de proceso en un recurso como $p^{max}{}_k = max._{j \in N}[p_{jk}]$ $\forall k \in M$ y el máximo de los tiempos mínimos como $p^{max^*} = max._{j \in N}[p^*_j]$. La fecha de entrega máxima es $d^{max} = max._{j \in N}[d_j]$. Se define z_{jk}como el TPI de la tarea$j$ en el recurso k y z^*_j como el TPI de la tarea j para el sistema en general donde $z_{jk} = d_j \times WD_k / d^{max} + p_{jk} \times WP_k / p^{max}{}_k$ mientras que $z^*_j = (d_j \times WD^* / d_{max}) + (p^*_j \times WP^* / p^{max*})$.

Heurística TR-ω

El objetivo de la heurística TR-ω es considerar todos los recursos de manera simultánea al crear la programación. La heurística tiene cuatro versiones, cada una considerando cuatro reglas para la selección de tareas.

Definición de las variables

N'	Conjunto de tareas pendientes para asignar a un recurso.
α	Tarea en consideración para asignación a los recursos en M.
ω	Define la regla a utilizarse para la selección de las tareas.
W	Conjunto de posibles programaciones.
w	Programa seleccionado.
N_k	Conjunto de las tareas asignadas al recurso k.
rc_k	El tiempo cuando el recurso k termina todas las tareas asignadas en el programa actual.

1. Definir$N' = N$.
2. Seleccionar la tareaα usando la reglaω. Definir $N' = N' –$ α.
   a. regla ω = p, tareaα = j: $min._{j \in N' k \in M}[p_{jk}]$
   b. regla ω = d, tareaα = j: $min._{j \in N'}[d_j]$.
   c. regla ω= e, tareaα = j: $min._{j \in N' k \in M}[e_{jk}]$.
   d. regla ω= z,tareaα = j: $min._{j \in N'}[z^*_j]$.
   Nota: En el caso de un empate se escoge entre éstas de manera aleatoria.
3. Definir$W$ como el conjunto de programaciones temporalmente generadas al: a) insertar la tareaα antes que cada una de las tareas en cada una de las secuencia originales de los m recursos y b) añadiendo la tareaα al final de la secuencia original de cada uno de los m recursos.
4. Seleccionar la programación w de W con el valor mínimo de $\sum_{j \in Nk, k \in M}[t_j]$. En el caso de empates se escoge la programación con menor $\sum_{k \in M} rc_k$. En el caso de empates se escoge la programación donde la tarea esté asignada a la máquina que tenga el menor e_{jk}.
5. Si $N' \neq \varnothing$ regresar al paso 3.
6. Fin de la heurística.

Heurística

La estrategia básica de la heurística *UR*-ω es considerar la asignación de tareas a un recurso a la vez. Esta heurística tiene dos fases; en la primera se asignan todos las tareas posibles al recurso seleccionado (con base en el tiempo total de proceso de las tareas sin asignar), con la restricción de que ninguna tarea asignada puede retrasarse. En la segunda fase se asignan aquellas tareas que no fueron asignadas en la fase 1, donde cada tarea es temporalmente insertada en cada posible posición del programa y de esas alternativas se

asigna a la posición/recurso donde el aumento en t_{ave} es menor.

Definición de las variables

L Conjunto de tareas consideradas y no asignadas.
M' Conjunto de recursos sin tareas asignadas.
P_k Suma de los tiempos de las tareas pendientes si son asignados al recurso k.
μ Recurso seleccionado por el proceso para asignación de tareas.
V_μ Conjunto de posibles secuencias en el recurso μ.
V_μ^* Conjunto de posibles secuencias en el recurso μque cumplen con restricción de β.
v^* Secuencia seleccionada.

Fase 1.

1. Definir$L = \varnothing, M' = M$ y$N' = N$.
2. Definir$P_k = \sum_{j \in N} p_{jk} \forall k \in M'$
3. Seleccionar el recurso $\mu = k$: $max._{k \in M'}[P_k]$ y definir $M' = M' - \mu$.
4. Seleccionar la tareaα usando la regla ω.Definir $N' = N' - \alpha$.
   a. regla ω = p, tareaα = j: $min._{j \in N'}[p_{j\mu}]$.
   b. regla ω = d, tareaα = j: $min._{j \in N'}[d_j]$.
   c. regla ω= e, tareaα = j: $min._{j \in N'}[e_{j\mu}]$.
   d. regla ω= z, tareaα = j: $min._{j \in N'}[z_{j\mu}]$.
   Nota: En el caso de un empate entre tareas, se escoge entre estas de manera aleatoria.
5. DefinirV_μ como el conjunto de secuencias temporalmente generadas al:a) insertar la tareaα antes que cada uno de las tareas en la secuencia original del recurso μ, y b) añadiendo la tareaα al final de la secuencia original del recursoμ.
6. DefinirV_μ^* como las secuencias en$V_\mu$donde$max._{j \in N\mu}[t_j] = 0$.
7. Si $V_\mu^* = \varnothing$, entonces definir $L = L + \alpha$ y regresar al paso 5.
8. Seleccionar v^* de V_μ^* con el valor mínimo de $\sum_{j \in N\mu}[t_j]$. En el caso de empates, seleccionar la secuencia con el valor mínimo de rc_μ.
9. Si $N' \neq \varnothing$ regresar al paso 5.
10. Si $M' \neq \varnothing$ y $L \neq \varnothing$ entonces se define $N' = L$ y regresar al paso 2.
11. Final de la fase 1. Si $L = \varnothing$ final de la heurística.

Fase 2

12. Definir$N' = L$.
13. Igual al paso 2 de la Heurística TR-ω
14. Igual al paso 3 de la Heurística TR-ω
15. Igual al paso 4 de la Heurística TR-ω
16. Si $N' \neq \varnothing$ regresar al paso 13.
17. Final de la heurística.

Ejemplo

La ilustración del problema y de las heurísticas propuestas se presenta en esta sección a través de la solución de un ejemplo. Se tienen dos recursos de producción (i.e. m = 2) y doce tareas por programar (i.e. n = 12). En la tabla 1 se presentan las características de las tareas: tiempo base de proceso, efecto de deterioro en cada recurso y el tiempo de entrega requerido. Por ejemplo, la tarea 2 puede ser procesada en 8 unidades de tiempo y en 2 unidades de tiempo en la máquina 1 y 2, respectivamente; el efecto de deterioro en la máquina 1 y 2 al realizar esta tarea es de 3,57% y 4,98%, respectivamente; finalmente la fecha de entrega de la tarea 2 es en la unidad de tiempo 9.

Tabla 1. Deterioro y tiempos de proceso y entrega para cada tarea

Tarea	Tiempo base de proceso		Efecto de deterioro (porcentaje)		Tiempo de entrega
j	p_{j1}	p_{j2}	e_{j1}	e_{j2}	d_j
1	8	9	5,38	1,26	18
2	8	2	3,57	4,98	9
3	3	2	9,68	6,89	2
4	8	5	1,49	5,24	6
5	1	1	4,28	9,66	11
6	6	6	1,04	7,72	10
7	6	7	2,46	2,90	12
8	8	9	5,95	4,36	14
9	2	7	7,64	8,86	10
10	6	5	3,44	6,31	5
11	2	3	9,54	8,90	8
12	2	2	8,79	3,77	2

Se presenta primero la heurística *TR* con ω = *e*. La tarea*j* = 6 es la de menor e_{jk}de entre todas las tareas (e_{j1} = 1,04%) por lo cual es la tarea seleccionada (α = 6). La tarea tiene una tardanza de 0 si es asignada a la posición 1 del recurso 1 (i.e. R1) o del recurso 2 (R2) – los dos posibles programas del conjunto *W*. Como la tarea tiene el mismo tiempo de proceso en ambos recursos, se asigna a R1, donde su efecto de deterioro es menor (paso no. 4). En el próximo ciclo se selecciona la tarea 1 (α = 1), ya que tiene el menor efecto de deterioro de las once tareas sin asignar. Dos de las posibles programaciones resultan en una tardanza total de 0; programa 1: (R1: 6, R2: 1); programa 2: (R1:6-1, R2:-). En el Programa 1, rc_1 = 6 y rc_2 = 9 (suma 15), mientras que en el programa 2, rc_1 = 6 + 8/(1 – 1,04%) = 14,084 y rc_2 = 0 (suma 14,084) por lo cual el programa 2 es seleccionado. En próximo ciclo se selecciona la tarea 4 (α = 4). Todos los posibles programas donde la tarea está asignada a R1, resultan en un tiempo de tardanza mayor que 0, mientras que el programa donde se le asigna a R2 tiene una tardanza de 0; por lo tanto, la tarea 4 se asigna a R2. La tabla 2 resume la información discutida sobre los primeros 3 ciclos del heurístico, y adicionalmente presenta los próximos dos ciclos (α = 7) y (α = 10) para completar la demostración. La programación final del recurso 1 es R1: 10-6-1; y la del recurso 2 es R2: 4-7.

Tabla 2. Primeros 5 ciclos de la heurística *TR-e*

α	*W*	t_{sum}	rc_1	rc_2	*w*
6	R1: 6; R2: - R1: - ; R2: 6	0 0	6 0	0 6	R1: 6; R2: -
1	R1: 1-6; R2: - R1: 6-1; R2: - R1: 6; R2: 1	4,34 **0** **0**	14,34 **14,08** 6	6 **0** 9	R1: 6-1; R2: -
4	R1: 4-6-1; R2: - R1: 6-4-1; R2: - R1: 6-1-4; R2: - R1: 6-1; R2: 4	10,39 12,37 16,63 **0**	22,30 22,29 22,63 14,08	0 0 0 5	R1: 6-1; R2: 4
7	R1: 7-6-1; R2: 4 R1: 6-7-1; R2: 4 R1: 6-1-7; R2: 4 R1: 6-1; R2: 7-4 R1: 6-1; R2: 4-7	4,59 2,41 8,49 6,15 **0,39**	20,44 20,35 20,49 14,08 14,08	5 5 5 12,15 12,39	R1: 6-1; R2: 4-7
10	R1: 10-6-1; R2: 4-7 R1: 6-10-1; R2: 4-7 R1: 6-1-10; R2: 4-7 R1: 6-1; R2: 10-4-7 R1: 6-1; R2: 4-10-7 R1: 6-1; R2: 4-7-10	**6,19** 9,89 15,88 10,56 11,44 13,21	20,59 20,44 20,49 14,08 14,08 14,08	12,39 12,39 12,39 18,22 18,16 17,82	R1: 10-6-1; R2: 4-7

Como segundo ejemplo de implementación de los métodos de solución se presenta la heurística *UR*-ω con ω = z. El primer elemento de esta heurística es la selección del recurso a programar. Se selecciona al recurso 1, (μ = 1) dado a que $P_1 > P_2$(P_1 = 60, P_2 = 58). Para determinar los valores de z_{j1}, se calcula P_1 y D, (P_1 = 60, D = 107) lo que resulta en $NC_1 = n \times P_k / (m \times D)$ = 12×60/ (2 × 107) = 3,36. El peso de las fechas de entrega es $WD_1 = max$. [0, *min.* [0,5 + (3 – 3,36)/ 3,36, 1]] = 0,3917 y el peso para los tiempos de proceso es WP_2 = 0,6083. El tiempo de proceso máximo en el recurso 1 es 8 y la fecha de entrega máxima es 18, por lo que z_{1k} = 8/8 × 0,39167 + 18/18 × 0,60833 = 1, z_{2k} = 8/8 × 0,39167 + 9/18 × 0,60833 = 0,69584, z_{3k} = 3/8 × 0,39167 + 2/18 × 0,60833 = 0,27164, y así sucesivamente.

De todas las tareas disponibles, el número 12 (α = 12) es el de menor z_{j1}. Al asignarse esta tarea a la primera posición del recurso 1, se obtiene una tardanza de 0, por lo cual se acepta esta secuencia. En el próximo ciclo de la fase 1 se selecciona la tarea 3 (α = 3) y el conjuntoW_1 tiene dos secuencias, secuencia 1: R1: 12-3 y secuencia 2: R1: 3-12. Ambos programas tienen tardanzas máximas mayores que 0, por lo cual ninguna de ellas es seleccionada y la tarea 3 es asignada al conjunto *L*. En el próximo ciclo de la fase 1 se selecciona la tarea 5 (α = 5) y el conjunto W_1 tiene dos secuencias, secuencia 1: R1: 12-5 y secuencia 2: R1: 5-12. De estas, la secuencia R1: 12-5 tiene una tardanza máxima de 0, por lo cual es seleccionada (paso no. 8). En la tabla 3 se presentan varios ciclos adicionales de esta heurística con μ = 1. Se puede apreciar que después de seis ciclos, la secuencia seleccionada (*v**) es 12-9-5-11 y el conjunto de tareas consideradas y no asignadas *L*= {3,10}.

Tabla 3. Primeros 6 ciclos de la heurística *UR-z*

α	V_2	V^*_2	v^*	L
12	12	12	12	{ - }
3	12-3 3-12			{3}
5	5-12 12-5	12-5	12-5	{3}
11	11-12-5 12-11-5 12-5-11	12-11-5 (rc_1 = 5.40) 12-5-11 (rc_1 = 5.39)	12-5-11	{3}
9	9-12-5-11 12-9-5-11 12-5-9-11 12-5-11-9	12-9-5-11 (rc_1 = 7.86) 12-5-9-11 (rc_1 = 7.87) 12-5-11-9 (rc_1 = 7.92)	12-9-5-11	{3}
10	10-12-9-5-11 12-10-9-5-11 12-9-10-5-11 12-9-5-10-11 12-9-5-11-10			{3, 10}

En el caso del recurso 1, ninguno de los ciclos posteriores a los presentados en la tabla 2 resulta en una secuencia con tardanza máxima igual a cero, por lo que la secuencia en el recurso 1 es R1: 12-9-5-11 para la fase 1. Para el recurso 2, la secuencia que resulta de la fase 1 es R2: 3-2-1. De esta fase el conjunto de tareas no asignadas es L = {4, 6, 7, 8, 10}. Dado que la fase 2 de la heurística *UR* es similar a la heurística *TR,* no detallamos su implementación en esta sección.

La figura 1 presenta la programación de todas las tareas que resulta de implementar las heurísticas *TR-e* y *UR-z*, con una tardanza promedio t_{ave} igual a 5,74 y 6,28, respectivamente. Se presentan los programas por recursos, incluyendo la secuencia de las tareas; al final de cada tarea se especifica el nivel de rendimiento del recurso (se indica arriba de la tarea que se completa), y la tardanza de cada tarea (se indica debajo de cada tarea). Es interesante notar cómo en el programa generado por *TR-e,* diez de las doce tareas tienen una tardanza mayor a 0, mientras que en el programa generado por *UR-z* sólo cinco de las doce tareas tienen una tardanza mayor a 0; sin embargo, el programa *UR-z* tiene una tardanza total mayor a la del programa *TR-e*. Estos resultados claramente indican diferencias entre los programas que se traducen en ventajas y desventajas que directamente impactan el nivel de servicio al cliente.

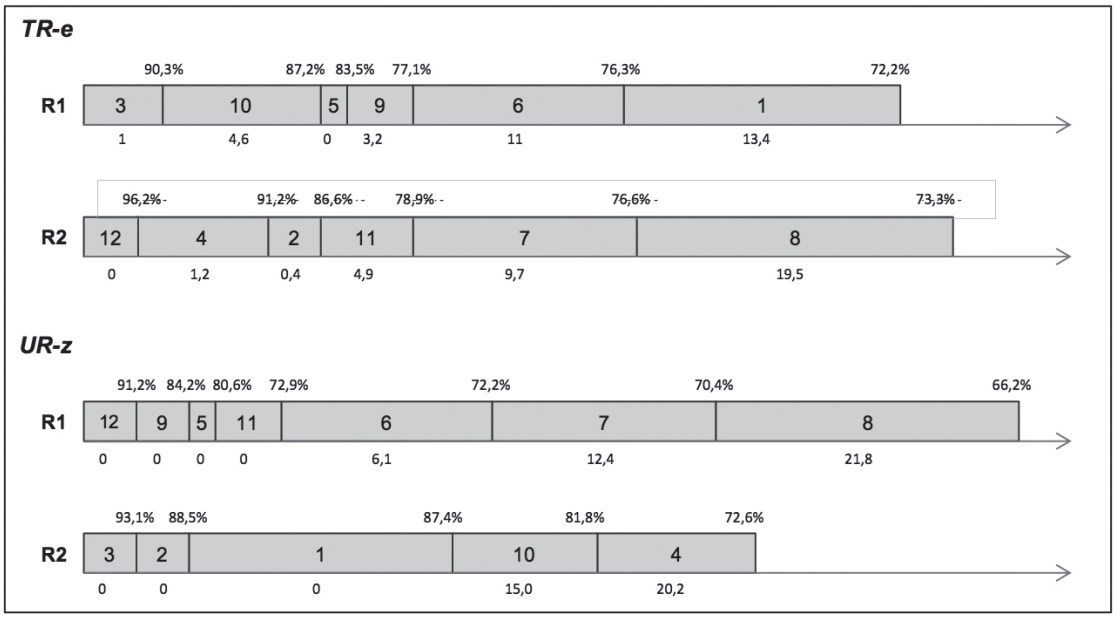


Figura 1. Programasque resultan de implementar las heurísticas *TR-e* y*UR-z*

Configuración de experimentos

Esta sección describe los experimentos utilizados para evaluar la eficacia de las heurísticas desarrolladas, considerando diversos factores operacionales:número de recursos, número de tareas que se determina basado en una proporción de tareas por recurso, rango de deterioro, y estrechez de las fechas de entrega.Estos factores han sido ampliamente investigados en múltiples estudios de programación de tareas; como referencia, el lector puede revisar los trabajos de Ho and Chang (1991) y Ruiz-Torres *et al*. (2013).

Los distintos niveles de los factores experimentales se describen a continuación. El número de recursos *m* se evalúa en tres niveles: 5, 10, y 20 recursos. También se consideran tres niveles de la proporción del número de tareas por número de recursos (*n/m*): 4, 7 y 10, donde el número de tareas *n* se determina simplemente por *m* × *n/m*; es decir, en un caso con *m* = 10 recursos y *n/m* = 7, el número de tareas *n* es 70. El rango del factor de deterioro (denominado *det*) se considera a dos niveles: *bajo* y *alto*. En el caso del nivel *bajo,* cada valor de e_{jk} se genera aleatoriamente por una variable uniforme con rango de 0% a 5%. En el nivel *alto* cada valor de e_{jk} se genera por una variable uniforme con rango de 5% a 10%.

La estrechez de las fechas de entrega se relaciona con cuánto tiempo tienen las tareas para cumplir con el requisito del cliente. Por ejemplo, un conjunto de tres tareas con tiempos de proceso: 5, 12, y 7 y con fechas de entrega: 6, 13 y 10, es mucho más estrecho que un conjunto de tareas con los mismos tiempos de proceso, pero con fechas de entrega de 8, 20, y 30. Claramente, en el primer caso, si se programan estas tres tareas en un solo recurso,habría por lo menos dos tareas con tardanzas, mientras que en el segundo caso existen múltiples programas con tardanza 0.

El tiempo de proceso base de la tarea *j* en el recurso *k* se genera con una variable aleatoria uniforme con rango de 1 a 99; $p_{jk} = U[1,99] \forall j \in N, \forall k \in M$.Para cada tarea se define $p^{min}_j = \min_{k \in M} p_{jk}$ y p^{avemin} como el promedio de los tiempos base mínimo, $p^{avemin} = 1/n \times \sum_{j \in N} p^{min}_j$. Se define D^* como la fecha de entrega máxima y $D^* = p^{avemin} \times n / (m \times est)$, donde *es t* es un valor cuantitativo

asociado al nivel de estrechez; mientras *est* aumenta, D^* es menor. Las fechas de entrega se estiman de la siguiente forma: para cada tarea se genera una variable aleatoria uniforme con rango de 0 a D^* y esta variable se suma al tiempo de proceso base mínimo de cada tarea para obtener la fecha de entrega: $d_j = min._{k \in M} p_{jk} + U(0, D^*)$. El valor de *es t* se evalúa en tres niveles: 2, 3, y 4. Hay entonces 3 ×2× 3 × 3 niveles experimentales y para cada uno se generan cincuenta repeticiones, lo cual representa un total de 2700 casos analizados.

RESULTADOS

Los resultados de los experimentos se evaluaron basados en tres medidas de desempeño: tardanza promedio por tarea, el error de la heurística contra la mejor solución obtenida, y el porcentaje de ocasiones donde una heurística obtuvo la menor tardanza promedio.El error estimado para cada problema se basa en el mejor resultado de las ocho versiones de la heurística:

$t^*_{ave} = min.\ [t_{ave}(TR\text{-}d), t_{ave}(UR\text{-}d), \ldots, t_{ave}(UR\text{-}z)]$ y $error\ (heurística) = t_{ave}(heurística)/t^*_{ave} - 1$.

El porcentaje de ocasiones donde una heurística obtiene el mejor resultado, caracteriza en cuantas ocasiones de las 50 repeticiones cada una de las heurísticas generó t^*_{ave}. Debido a la posibilidad de que más de una heurística genere una programación con exactamente el mismo valor de t_{ave}, es posible que exista más de una heurística "ganadora" para cada repetición.

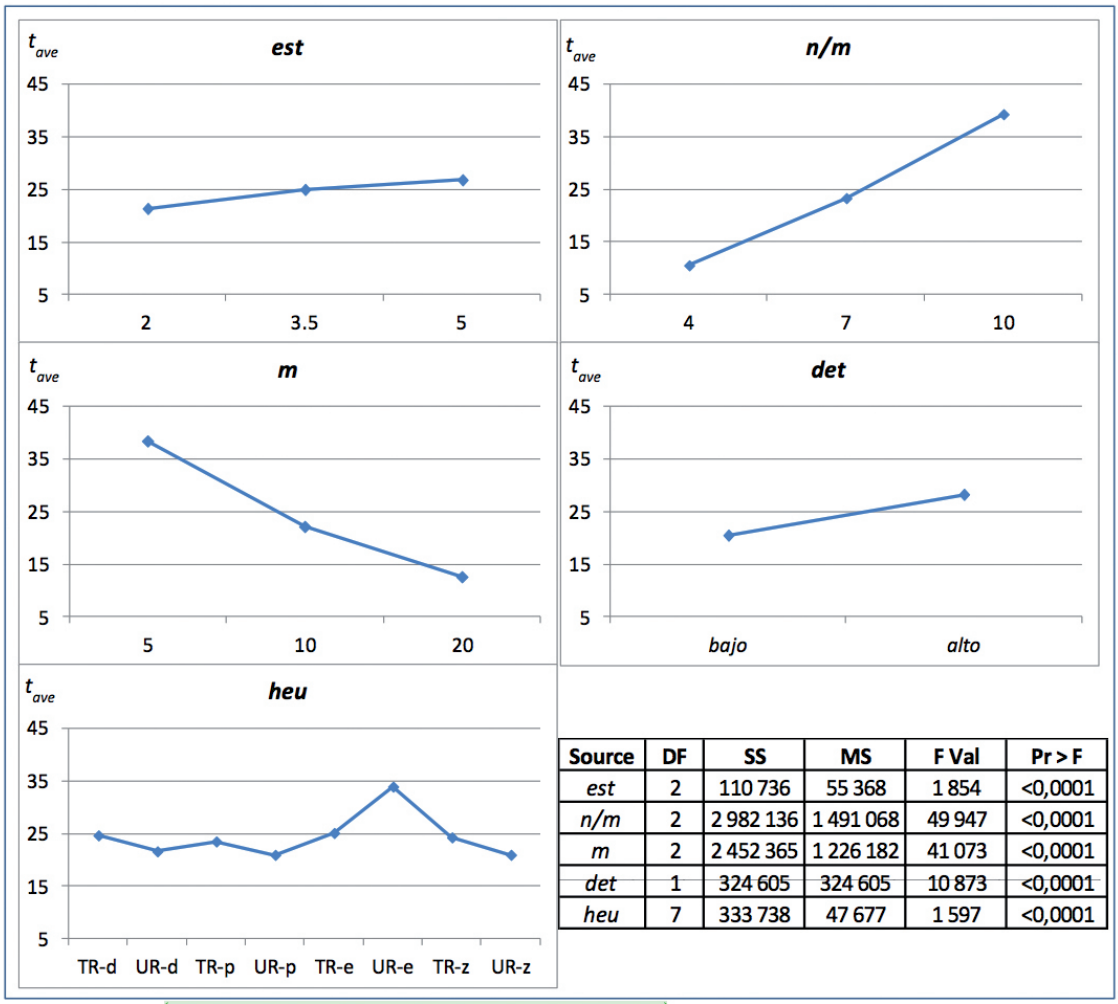
Resultados: Tardanza promedio

La tabla 4 presenta los resultados de $t_{ave} = t_{sum}/n$ para cada una de las condiciones experimentales. Los resultados en "negritas" indican el valor menor de la tardanza promedio para esa combinación de condiciones. Cuatro de las ocho heurísticas obtuvieron el valor mínimo en por lo menos una combinación experimental: *UR-d*, *TR-p*, *UR-p*, y *UR-z*. La heurística que obtuvo el mayor número de valores mínimos de la tardanza promedio fue *UR-p* con 25, seguida por *UR-z* con 18.

La figura 2 presenta los efectos que tienen los factores experimentales en t_{ave}(el eje verticales t_{ave}) y los resultados de un análisis de varianza (ANOVA por sus siglas en inglés). Como se puede observar en las gráficas y en los resultados del ANOVA, los factores *m* y *n/m* son los más significativos, aunque todos los factores principales son significativos. Todos los resultados presentados en las gráficas son intuitivos: al aumentar el número de recursos (*m*), la tardanza promedio baja; mientras que al aumentar la relación de tareas al número de recursos (*n/m*), la deterioración (*det*), y la estrechez (*est*), la tardanza promedio aumenta. Respecto a la eficacia de las heurísticas (*heu*), se observa que *UR-e* generó el valor más grande de la tardanza promedio, mientras que *UR-d*, *UR-p* y UR-z generaron los valores más bajos. Notamos aquí que, aunque sólo se presentan en la tabla de ANOVA los efectos principales, las interacciones fueron incluidas en el modelo y son significativas. El modelo alcanzó un valor de R^2 = 91,8%.

Tabla 4. Resultados de t_{ave}

est	*det*	*m*	*n/m*	*TR-d*	*UR-d*	*TR-p*	*UR-p*	*TR-e*	*UR-e*	*TR-z*	*UR-z*
2	*bajo*	5	4	12,1	12,4	**10,4**	12,4	11,9	19,4	11,3	12,3
			7	25,6	25,0	**24,1**	24,9	27,0	46,0	26,7	25,0
			10	40,0	38,2	38,7	**36,7**	42,3	74,2	42,8	36,9
		10	4	6,7	7,0	**6,1**	6,9	6,6	11,6	6,7	6,8
			7	14,3	13,6	13,0	**12,8**	15,0	25,6	14,4	13,0
			10	23,8	21,3	21,8	**20,0**	24,1	42,3	24,6	20,2
		20	4	3,8	3,7	**3,4**	3,7	3,8	6,3	3,8	3,7
			7	8,2	7,4	7,4	**7,0**	8,1	13,6	8,2	7,1
			10	14,2	12,2	12,6	**11,5**	14,6	24,2	14,4	11,6
	alto	5	4	16,1	15,3	**14,6**	15,1	16,2	24,3	16,5	15,3
			7	36,2	33,9	34,3	**32,8**	37,3	54,4	35,0	33,1
			10	65,2	57,9	60,7	55,4	67,6	95,1	65,1	**55,4**
		10	4	8,5	8,5	**8,1**	8,3	9,1	12,8	8,6	8,4
			7	21,3	18,4	19,1	**17,5**	20,9	30,5	21,1	17,7
			10	39,1	32,9	35,8	**31,6**	40,5	56,1	39,4	31,7
		20	4	4,9	4,7	4,6	4,6	5,1	7,2	4,9	**4,6**
			7	12,0	10,3	10,8	**9,9**	12,2	17,1	12,0	10,0
			10	22,0	18,2	20,1	**17,3**	22,6	30,8	22,1	17,4
3,5	*bajo*	5	4	14,2	14,0	14,3	**13,9**	14,8	23,4	14,4	14,0
			7	32,9	29,8	32,1	29,1	33,0	47,4	31,5	**28,8**
			10	51,3	45,7	50,0	**44,1**	53,1	73,1	51,4	44,2
		10	4	8,7	8,2	8,3	8,1	8,9	13,1	8,5	**8,0**
			7	18,7	16,5	17,9	16,2	18,7	26,9	18,3	**16,1**
			10	30,7	25,2	28,4	24,1	30,1	42,7	29,6	**23,9**
		20	4	5,1	4,7	5,0	4,7	5,3	7,3	5,0	**4,7**
			7	10,8	9,3	10,3	9,0	11,0	15,3	10,7	**9,0**
			10	17,7	14,7	16,4	**13,6**	17,6	23,8	17,2	13,7
	alto	5	4	16,8	**16,3**	16,3	16,5	17,0	23,6	16,5	16,3
			7	41,8	37,6	41,0	**36,0**	44,0	54,0	40,9	36,1
			10	78,1	68,5	74,8	**65,4**	80,8	99,4	77,6	65,6
		10	4	10,8	9,9	10,5	9,9	11,0	14,2	10,8	**9,8**
			7	25,7	22,0	24,5	**21,0**	27,0	33,0	25,5	21,2
			10	46,7	38,6	44,4	**36,8**	46,8	54,8	44,9	36,9
		20	4	6,2	5,8	6,1	**5,6**	6,6	8,4	6,2	5,7
			7	14,8	12,3	13,9	11,9	15,0	18,3	14,4	**11,9**
			10	26,4	21,2	25,1	20,1	26,5	30,3	25,6	**20,0**
5	*bajo*	5	4	16,0	15,0	15,2	**14,8**	16,5	22,4	15,8	14,8
			7	33,7	30,0	33,4	29,4	33,3	47,6	33,3	**29,0**
			10	58,9	51,1	56,5	**49,0**	58,0	77,9	55,8	49,1
		10	4	9,5	**8,6**	9,6	8,7	10,0	13,9	9,2	8,6
			7	21,0	17,8	20,6	17,6	20,8	28,1	20,5	**17,5**
			10	33,5	27,6	32,1	26,6	33,1	43,2	32,3	**26,5**
		20	4	5,8	5,2	5,6	**5,2**	5,9	8,0	5,7	5,3
			7	11,9	10,1	11,3	**9,7**	11,7	15,6	11,5	9,8
			10	19,4	15,7	18,5	15,2	19,1	24,8	18,8	**15,1**
	alto	5	4	19,0	**17,8**	19,0	18,1	19,8	26,4	19,0	17,9
			7	49,7	**42,0**	48,4	42,5	50,2	61,3	47,5	42,3
			10	80,4	69,0	78,7	66,7	83,7	97,4	78,2	**66,3**
		10	4	12,4	**10,7**	11,7	10,8	13,0	16,3	12,0	10,8
			7	27,6	23,2	26,7	22,6	27,9	32,7	26,8	**22,6**
			10	49,7	39,6	47,0	**38,4**	48,4	55,6	47,6	38,7
		20	4	6,9	6,1	6,6	**6,0**	7,2	8,9	6,8	6,1
			7	16,1	13,3	15,5	13,1	16,4	19,2	15,7	**12,9**
			10	28,1	22,4	26,6	**21,8**	28,7	31,7	27,3	22,0



Source	DF	SS	MS	F Val	Pr > F
est	2	110 736	55 368	1 854	<0,0001
n/m	2	2 982 136	1 491 068	49 947	<0,0001
m	2	2 452 365	1 226 182	41 073	<0,0001
det	1	324 605	324 605	10 873	<0,0001
heu	7	333 738	47 677	1 597	<0,0001

Figura 2. Efectos de los factores experimentales y resultados de ANOVA para t_{ave}

Resultados: Error relativo a la mejor solución

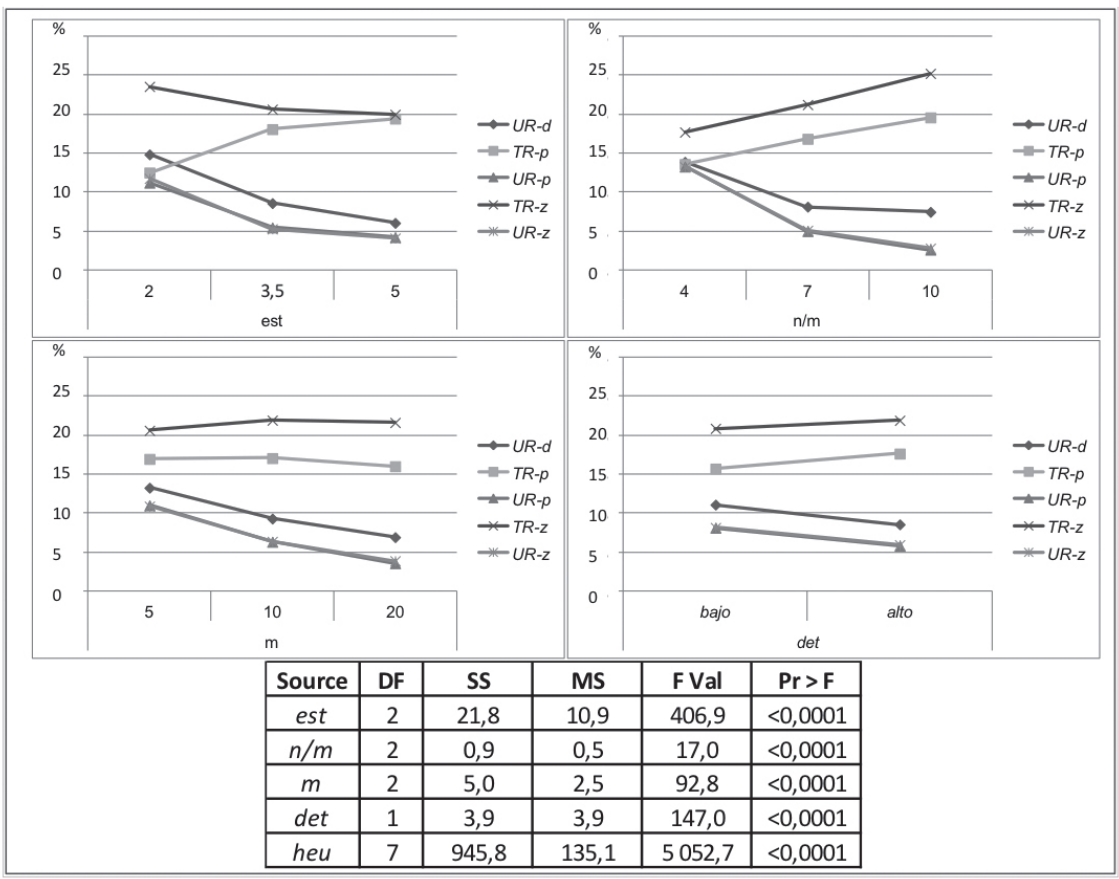
La tabla 5 presenta los resultados del error relativo a la mejor solución obtenida para cada heurística y para cada una de las condiciones experimentales; los resultados en "negritas" indican el menor valor del error relativo para esa combinación de factores experimentales (se utilizan los mismos factores que para t_{ave}). De los 2700 experimentos, las heurísticas *UR-p* y *UR-z* fueron las de menor error con un promedio de 7,0% y 7,1%, respectivamente. Los resultados generales para las otras seis heurísticas son *UR-d*: 9,8%, *TR-p*: 16,7%, *TR-z*: 21,4%, *TR-d*: 23,1%, *TR-e*: 25,5% y *UR-e*: 75,6%.

ISSN 0717-9103
ISSN Online 0718-8307
Universidad del Bío-Bío

Tabla 5. Resultados del error relativo a la mejor solución

est	*det*	*m*	*n/m*	*TR-d (%)*	*UR-d (%)*	*TR-p (%)*	*UR-p (%)*	*TR-e (%)*	*UR-e (%)*	*TR-z (%)*	*UR-z (%)*
2	*bajo*	5	4	31,9	41,8	**11,3**	40,9	30,5	124,8	24,3	40,3
			7	19,5	17,2	**11,8**	15,4	27,6	120,0	25,0	16,4
			10	18,8	12,8	13,3	**7,9**	24,6	119,9	25,5	8,4
		10	4	22,7	30,2	**12,1**	28,7	21,6	119,7	22,7	27,9
			7	20,1	14,2	9,3	**8,0**	26,3	115,4	21,0	9,4
			10	23,4	10,2	12,8	**3,4**	25,4	119,5	27,5	4,4
		20	4	17,2	15,3	**5,9**	16,1	16,9	96,6	16,4	15,1
			7	21,3	8,9	9,0	**3,4**	19,3	102,3	22,1	4,7
			10	25,5	8,1	11,3	**1,4**	28,4	114,1	26,8	2,4
	alto	5	4	26,2	25,3	**16,0**	23,8	31,8	103,1	28,6	25,2
			7	23,1	15,7	16,3	**12,6**	27,2	86,3	18,6	13,3
			10	22,8	9,4	15,0	4,6	27,8	80,8	22,8	**4,7**
		10	4	17,5	17,8	**12,0**	14,8	24,9	77,6	18,6	16,6
			7	27,2	9,4	14,0	**4,4**	24,8	83,2	25,5	5,5
			10	27,6	7,4	17,0	**3,2**	32,4	84,2	29,0	3,7
		20	4	14,4	9,9	8,3	8,6	18,2	70,2	14,8	**7,8**
			7	24,0	6,6	11,8	**2,8**	26,5	77,3	24,4	3,7
			10	29,2	6,9	18,2	**1,8**	32,9	81,1	30,0	2,2
3,5	*bajo*	5	4	18,0	16,7	19,2	**15,6**	22,3	99,2	18,2	16,5
			7	20,7	8,8	18,1	7,4	21,9	77,4	15,8	**6,1**
			10	21,4	8,4	18,2	**4,0**	26,0	74,4	21,8	4,0
		10	4	18,5	11,9	13,4	10,3	21,2	81,1	16,9	**9,8**
			7	20,2	6,1	15,3	4,4	20,7	74,4	17,4	**3,4**
			10	30,6	7,3	20,9	2,4	28,1	82,5	26,0	**1,6**
		20	4	13,9	6,1	12,2	6,4	19,7	66,0	12,9	**5,9**
			7	23,0	5,9	17,0	2,6	25,0	74,3	21,4	**2,5**
			10	31,1	8,7	21,3	**0,6**	30,2	75,7	26,9	1,4
	alto	5	4	18,3	**15,9**	14,9	17,3	21,7	70,2	17,5	15,8
			7	21,8	9,2	19,1	**4,8**	28,1	57,8	18,6	4,7
			10	23,5	8,1	18,5	**2,8**	27,9	57,9	22,8	3,1
		10	4	19,7	8,9	16,4	9,4	20,6	57,5	18,8	**8,2**
			7	24,5	6,7	18,5	**1,8**	31,2	60,0	23,9	2,8
			10	29,9	7,3	23,4	**2,0**	30,3	52,9	24,9	2,3
		20	4	15,2	6,2	13,3	**3,5**	21,6	56,0	14,6	4,6
			7	27,1	5,8	19,1	2,2	28,3	57,0	23,5	**1,9**
			10	32,9	7,0	26,4	1,5	33,6	52,8	29,0	**0,9**
5	*bajo*	5	4	21,6	15,2	16,0	**14,7**	26,4	75,1	20,4	14,6
			7	21,7	7,5	20,4	5,1	19,4	71,6	19,7	**3,9**
			10	23,2	6,8	18,3	**2,6**	21,8	63,6	17,1	2,8
		10	4	15,9	**5,3**	17,4	6,4	22,0	72,2	12,7	5,4
			7	23,1	4,1	20,8	3,2	21,7	65,5	20,5	**2,9**
			10	28,8	5,7	23,2	2,3	27,0	66,4	24,1	**1,5**
		20	4	17,2	5,3	13,2	**4,1**	18,2	62,2	13,6	6,0
			7	24,7	6,0	18,3	**1,2**	22,2	63,8	20,1	2,1
			10	30,0	5,3	24,3	1,7	28,2	66,3	26,0	**1,1**
	alto	5	4	15,4	**8,7**	15,7	10,2	21,8	64,1	15,1	9,0
			7	24,4	**4,9**	21,1	5,8	25,4	53,2	18,4	5,2
			10	23,7	6,4	21,7	2,6	29,4	50,4	20,6	**2,1**
		10	4	20,2	**4,9**	14,5	5,4	27,1	59,9	16,3	5,0
			7	25,2	5,4	21,1	2,6	27,1	49,0	22,0	**2,2**
			10	31,7	5,0	24,7	**1,6**	28,3	47,7	26,2	2,5
		20	4	18,2	4,6	12,9	**2,8**	22,8	51,7	16,1	4,7
			7	27,0	4,6	22,0	2,9	29,8	51,7	23,6	**1,6**
			10	30,7	4,0	23,8	**1,4**	33,5	47,5	26,6	2,1

De estos resultados se determina fácilmente que *UR-e* tiene un desempeño muy pobre y no se debe considerar en el resto de los análisis. Nótese que en la medida de desempeño anterior, *UR-e* generó el valor más grande de la tardanza promedio; por tanto, *UR-e* comprueba su bajo desempeño en ambas medidas para estimar la eficacia de la heurística. En la figura 3 se presenta el resultado de ANOVA y la interacción entre las heurísticas y los cuatro factores experimentales (el eje verticales el porcentaje de *error*). Para mejor claridad, en las gráficas no se incluyen las tres heurísticas de peor desempeño (*TR-d*, *TR-e*, y *UR-e*); además se nota que *UR-z* y *UR-p* son en todos los casos casi idénticas (por lo que no hay notable diferencia en las gráficas). Los resultados de ANOVA demuestran que en términos del *error*, el factor heurística es el más significativo,que es lo deseado, ya que demuestra que no es el ambiente (los factores operacionales) lo que determina el error. En el ANOVA las interacciones de los factores operacionales con la heurística son también significativas y el R^2 de este ANOVA es 65,94%.



Source	DF	SS	MS	F Val	Pr > F
est	2	21,8	10,9	406,9	<0,0001
n/m	2	0,9	0,5	17,0	<0,0001
m	2	5,0	2,5	92,8	<0,0001
det	1	3,9	3,9	147,0	<0,0001
heu	7	945,8	135,1	5 052,7	<0,0001

Figura 3. Efectos de los factores experimentales y resultados de ANOVA en el *error relativo a la mejor solución.*

Las relaciones resultantes entre las heurísticas y los factores experimentales son interesantes, en particular cuando no son semejantes para todas las heurísticas. En el caso del factor *est*, para cuatro de las heurísticas el error disminuye a medida que las fechas de entrega son más estrechas, mientras que aumentan para una (*TR-p*), lo que puede indicar que *TR-p* podría ejecutar muy bien cuando *est*< 2 (o sea menos estrechez en las fechas de entrega). En el caso del factor *det*, el desempeño de las dos heurísticas *TR* empeoran ligeramente al aumentar el nivel general de deterioro, mientras que las heurísticas *UR* mejoran ligeramente. Para el factor

m, las heurísticas *TR* no cambian de manera notable, mientras que para las heurísticas *UR*, el error disminuye a medida que el número de recursos aumenta. Finalmente, para el factor *n/m* el desempeño de las heurísticas *TR* empeora a medida que *n/m* aumenta mientras que mejora para las *UR*. Todo indica a que más complejidad operacional (mayor *m*, *n/m*, *est* y *det*), las heurísticas *UR-z* y *UR-p*se desempeñan bien relativamente al resto de las versiones.

Resultados: Porcentaje de mejores soluciones

La tabla 6 presenta los resultados del promedio de las mejores soluciones generadas por cada una de las heurísticas y la suma de los porcentajes. En la tabla 6 no se incluyó *TR-z,* dado que esta heurística nunca generó una mejor solución.Los resultados en "negritas" indican el valor mayor para esa combinación, notando que no son los mismos puntos que en las tablas anteriores y que existen varios puntos donde dos heurísticas generan el mismo porcentaje de mejores soluciones.

Tabla 6. Resultados del *porcentaje de mejores soluciones*

est	*det*	*m*	*n/m*	*TR-d (%)*	*UR-d (%)*	*TR-p (%)*	*UR-p (%)*	*TR-e (%)*	*TR-z (%)*	*UR-z (%)*	*Sum (%)*
2	*bajo*	5	4	18	8	**46**	6	14	20	10	122
			7	16	16	**32**	10	8	12	10	104
			10	18	10	**30**	20	4	8	10	100
		10	4	14	6	**42**	6	18	18	8	112
			7	6	10	**34**	20	**4**	12	14	100
			10	8	6	16	**42**	0	2	26	100
		20	4	14	14	**40**	6	16	8	6	104
			7	0	20	20	**36**	4	4	16	100
			10	0	16	2	**56**	0	0	26	100
	alto	5	4	8	16	**34**	16	28	8	16	126
			7	6	12	**26**	18	6	18	14	100
			10	4	20	8	24	0	4	**40**	100
		10	4	20	10	**28**	16	18	14	2	108
			7	4	16	4	**34**	4	6	32	100
			10	0	16	8	**38**	0	0	**38**	100
		20	4	8	10	**28**	6	16	18	14	100
			7	0	22	4	**40**	0	0	34	100
			10	0	12	0	**50**	0	0	38	100
3,5	*bajo*	5	4	20	8	16	**24**	24	16	16	124
			7	4	**28**	6	18	6	12	**28**	102
			10	2	24	2	**42**	2	4	26	102
		10	4	12	16	**20**	18	10	14	16	106
			7	6	22	8	14	2	6	**42**	100
			10	0	16	0	40	0	2	**42**	100
		20	4	4	**28**	12	28	4	12	12	100
			7	0	26	4	32	0	0	**38**	100
			10	0	2	0	**64**	0	0	34	100
	alto	5	4	20	18	**24**	10	24	24	10	130
			7	4	8	2	**46**	6	6	34	106
			10	6	14	2	**40**	0	0	38	100
		10	4	14	**22**	12	14	14	12	14	102
			7	0	10	6	**50**	0	2	32	100
			10	0	12	0	**52**	0	0	36	100

		20	4	10	18	8	**36**	4	4	20	100
			7	0	18	0	**44**	0	0	38	100
			10	0%	6	0	36	0	0	**58**	100
5	*bajo*	5	4	22	22	**28**	18	12	22	16	140
			7	4	20	2	30	4	14	**46**	120
			10	2	28	2	**44**	2	0	32	110
		10	4	6	**42**	8	18	4	12	26	116
			7	0	28	2	20	6	2	**44**	102
			10	0	18	0	38	0	0	**44**	100
		20	4	8	**28**	12	22	2	14	14	100
			7	0	14	0	52	0	0	34	100
			10	0	12	0	32	0	0	**56**	100
	alto	5	4	20	**28**	12	20	16	24	12	132
			7	0	**44**	2	28	0	4	34	112
			10	0	22	0	36	0	0	**52**	110
		10	4	0	**40**	12	28	6	4	22	112
			7	0	20	0	34	2	0	**46**	102
			10	0	22	0	**46**	0	0	34	102
		20	4	4	24	10	**46**	0	6	12	102
			7	0	22	2	26	0	0	**50**	100
			10	0	22	0	**46**	0	0	32	100

Consistente con los resultados anteriores, las heurísticas *UR-p* y *UR-z* fueron las de mejor desempeño, generando para los 2700 problemas un 30,3% y 27,7% de las mejores soluciones, respectivamente. Como se mencionó anteriormente, estos porcentajes incluyen casos donde más de una heurística encontró el mismo resultado. Por ejemplo, en la primera línea (*m* = 5, *n/m* = 4, *est*= 2 y *det* = *bajo*) la suma de los porcentajes es 122%, por lo que se generaron 11 soluciones repetidas (mejores programas). Igualmente, hay múltiples combinaciones de los niveles experimentales donde cada mejor solución fue generada por sólo una heurística; por ejemplo, en la tercera línea de la tabla (*m* = 5, *n/m* = 10, *est* = 2 y*det* = bajo), donde los promedios suman a 100%. Para todos los 2700 problemas la suma de los promedios es 106%, por lo que relativamente existen pocos casos de múltiples heurísticas generando la mejor solución. Este resultado es también relevante, porque indica que ninguna de las heurísticas es absolutamente dominante y que el utilizar múltiples heurísticas resulta en la obtención de mejores programas de tareas.

La figura 4 presenta el efecto que tienen los factores operacionales en el desempeño de las heurísticas para la métrica de porcentaje de mejores soluciones. Las cuatro gráficas demuestran efectos diferentes para las heurísticas. En el caso del factor, *est*, el porcentaje de mejores soluciones generadas por *UR-p*, *UR-d* y *UR-z* incrementa cuando la estrechez incrementa, mientras que empeora para las otras cuatro heurísticas presentadas. Aquí es notable el deterioro de la heurística *TR-p,* que generó sobre 20% de las mejores soluciones cuando *est* = 2, pero se redujo a menos de 5% cuando *est*> 2. En el caso del factor *n/m*, el efecto es notable; cuando *n/m* = 4 todas las heurísticas "contribuyen" generando entre 10-23% de las mejores soluciones, mientras que cuando *n/m* = 10 sólo tres de las heurísticas están generando las mejores soluciones (*UR-p*, *UR-d*, y *UR-z*). El efecto de los factores *m* y *de t*es menos notable, al aumentar *m*, *UR-p* y *UR-z* mejoran en desempeño, mientras que las otras cinco empeoran; al incrementar *det*, *UR-p*, *UR-z* y *UR-d* mejoran ligeramente, mientras que el resto de las heurísticas empeoran. Claramente esto reitera que mientras la complejidad operacional aumenta (mayor *m*, *n/m*, *est* y *det*), las heurísticas *UR-p* y *UR-z* son las de mejor desempeño, generando la mayoría de las mejores soluciones.

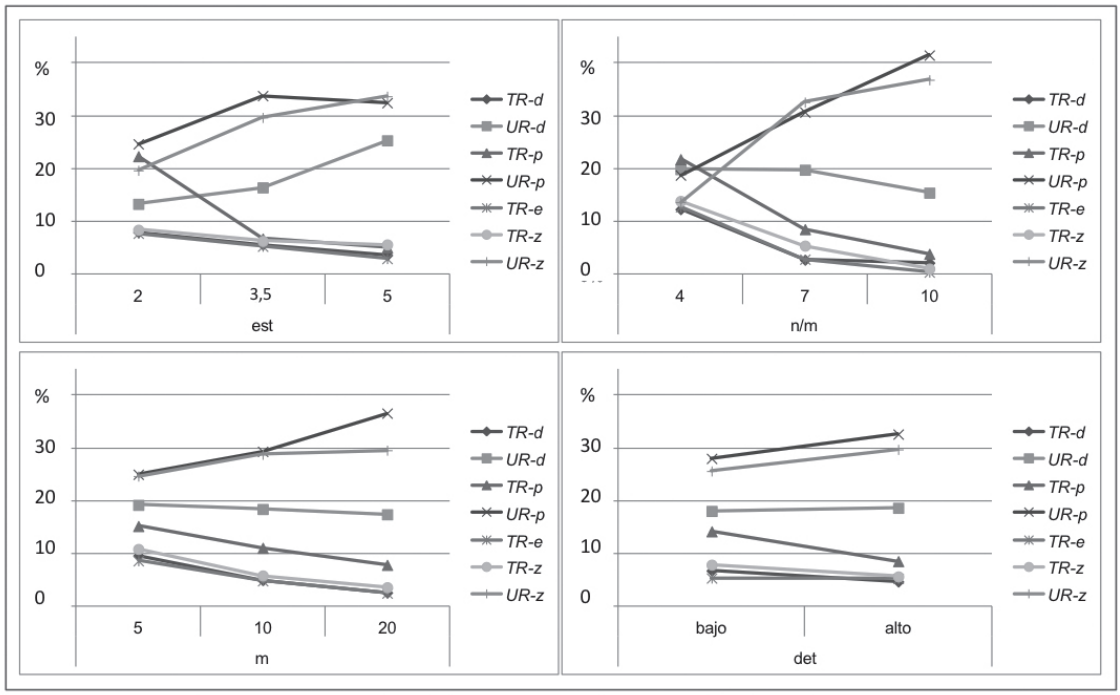


Figura 4. Efectos de los factores experimentales en el *promedio de mejores soluciones.*

CONCLUSIONES

La solución de la formulación matemática del problema es altamente compleja; por tanto, dos heurísticas fueron desarrolladas para generar buenas soluciones en poco tiempo de cómputo. Las heurísticas se aplicaron a pequeños problemas con fines de ilustración de la metodología, y para fines de investigación un total de 2700 casos han sido resueltos usando diferentes factores experimentales,para de esta forma realizar un análisis de sensibilidad del desempeño de las heurísticas ante tales factores. Los resultados mostraron que las heurísticas generan buenas soluciones y pueden ser usadas por los tomadores de decisiones, quienes frecuentemente enfrentan este problema de programación de tareas en recursos paralelos. Para el tomador de decisiones es de alta relevancia el hecho de que no se encontró ninguna heurística absolutamente dominante, lo cual sugiere el uso de múltiples heurísticas para un mismo problema, con el objetivo de identificar la mejor programación de tareas.

Investigaciones futuras inmediatas consideran el desarrollo de otros algoritmos, tales como meta-heurísticos, para resolver el modelo matemático presentado, y el desarrollo del modelo mismo con la incorporación de otros parámetros, tales como la probabilidad en los tiempos de proceso y en la calidad del producto, entre otros.

AGRADECIMIENTOS

Este trabajo científico ha sido financiado por el Proyecto Prometeo de la Secretaría Nacional de Ciencia, Tecnología e Innovación del gobierno de Ecuador.

BIBLIOGRAFIA

ALIDAEE, B., and WOMER, N.K. Scheduling with time dependent processing times: review and extensions. *Journal of the Operational Research Society*, 1999, vol. 50, no. 7, p. 711-720.

ANGHINOLFI, D., and PAOLUCCI, M. Parallel machine total tardiness scheduling with a new hybrid metaheuristic approach. *Computers & Operations Research*, 2007, vol. 34, no. 11, p. 3471-3490.

AZIZOGLU, M., and KIRCA, O. Tardiness minimization on parallel machines.*International Journal of Production Economics*, 1998, vol. 55, no. 2, p. 163-168.

BANK, J., and WERNER, F. Heuristic algorithms for unrelated parallel machine scheduling with a common due date, release dates, and linear earliness and tardiness penalties. *Mathematical and Computer Modelling*, 2001, vol. 33, no. 4, p. 363-383.

BILGE, Ü., KIRAÇ, F., KURTULAN, M., and PEKGÜN, P. A tabu search algorithm for parallel machine total tardiness problem. *Computers & Operations Research*, 2004, vol. 31, no. 3, p. 397-414.

BISKUP, D., and HERRMANN, J. Single-machine scheduling against due dates with past-sequence-dependent setup times. *European Journal of Operational Research*, 2008, vol. 191, no. 2, p. 587-592.

BISKUP, D., HERRMANN, J., and GUPTA, J.N. Scheduling identical parallel machines to minimize total tardiness. *International Journal of Production Economics*, 2008, vol. 115, no. 1, p. 134-142.

BROWNE, S., and YECHIALI, U. Scheduling deteriorating jobs on a single processor. *Operations Research*, 1990, vol. 38, no. 3, p. 495-498.

CHENG, T.E., DING, Q., and LIN, B.M. A concise survey of scheduling with time-dependent processing times. *European Journal of Operational Research*, 2004, vol. 152, no. 1, p. 1-13.

CHENG, T.E., HSU, C.J., HUANG, Y.C., and LEE, W.C. Single-machine scheduling with deteriorating jobs and setup times to minimize the maximum tardiness. *Computers & Operations Research*, 2011, vol. 38, no. 12, p. 1760-1765.

CHENG, M., TADIKAMALLA, P.R., SHANG, J., and ZHANG, S. Bicriteria hierarchical optimization of two-machine flow shop scheduling problem with time-dependent deteriorating jobs. *European Journal of Operational Research*, 2014, vol. 234, no. 3, p. 650-667.

DELLA CROCE, F., GARAIX, T., and GROSSO, A. Iterated local search and very large neighborhoods for the parallel-machines total tardiness problem.*Computers & Operations Research*, 2012, vol. 39, no. 6, p. 1213-1217.

EREN, T., and GÜNER, E. Minimizing total tardiness in a scheduling problem with a learning effect. *Applied Mathematical Modelling*, 2007, vol. 31, no. 7, p. 1351-1361.

GUPTA, J.N., and GUPTA, S. K. Single facility scheduling with nonlinear processing times. *Computers & Industrial Engineering*, 1988, vol. 14, no. 4, p. 387-393.

HO, J.C., and CHANG, Y.L. Heuristics for minimizing mean tardiness for m parallel machines. *Naval Research Logistics (NRL)*, 1991, vol. 38, no. 3, p. 367-381.

HUANG, X., and WANG, M. Z. Parallel identical machines scheduling with deteriorating jobs and total absolute differences penalties. *Applied Mathematical Modelling*, 2011, vol. 35, no. 3, p. 1349-1353.

JOO, C.M., and KIM, B.S. Genetic algorithms for single machine scheduling with time-dependent deterioration and rate-modifying activities.*Expert Systems with Applications*, 2013, vol. 40, no. 8,p. 3036–3043.

JOUGLET, A., and SAVOUREY, D. Dominance rules for the parallel machine total weighted tardiness scheduling problem with release dates. *Computers & Operations Research*, 2011, vol. 38, no. 9, p. 1259-1266.

LIAW, C.F., LIN, Y.K., CHENG, C.Y., and CHEN, M. Scheduling unrelated parallel machines to minimize total weighted tardiness. *Computers & Operations Research*, 2003, vol. 30, no. 12, p. 1777-1789.

LIU, M., ZHENG, F., WANG, S., and XU, Y. Approximation algorithms for parallel machine scheduling with linear deterioration. *Theoretical Computer Science*, 2013, vol. 497, no. 29, p.108–111.

MAZDEH, M. M., ZAERPOUR, F., ZAREEI, A., and HAJINEZHAD, A. Parallel machines scheduling to minimize job tardiness and machine deteriorating cost with deteriorating jobs. *Applied Mathematical Modelling*, 2010, vol. 34, no. 6, p. 1498-1510.

MOSHEIOV, G.A note: Multi-machine scheduling with general position-based deterioration to minimize total load. *International Journal of Production Economics*, 2012, vol. 135, no. 1, pp. 523-525.

RUIZ-TORRES, A. J., PALETTA, G., and PÉREZ, E. Parallel machine scheduling to minimize the makespan with sequence dependent deteriorating effects.*Computers & Operations Research*, 2013, vol. 40, no. 8, p. 2051–2061.

SHIM, S.O., and KIM, Y.D. Scheduling on parallel identical machines to minimize total tardiness. *European Journal of Operational Research*, 2007, vol. 177, no. 1, p. 135-146.

STERNA, M.A survey of scheduling problems with late work criteria.Omega, 2011, vol. 39, no. 2, p. 120-129.

SUN, H., and WANG, G. Parallel machine earliness and tardiness scheduling with proportional weights. *Computers & Operations Research*, 2003, vol. 30, no. 5, p. 801-808.

TANAKA, S., and ARAKI, M.A branch-and-bound algorithm with Lagrangian relaxation to minimize total tardiness on identical parallel machines.*International Journal of Production Economics*, 2008, vol. 113, no. 1, p. 446-458.

TOKSARI, M. D., and GÜNER, E. The common due-date early/tardy scheduling problem on a parallelmachine under the effects of time-dependent learning and linear and nonlinear deterioration. *Expert Systems with Applications*, 2010, vol. 37, no. 1, p. 92-112.

VELEZ, M., and LÓPEZ, J.A. Variable neighborhood search algorithm to minimize the total weighted tardiness on a batch processing machine.*Revista Ingenieria Industrial*, 2011, vol. 10, no. 1, p 5-18.

WU, C. C., CHENG, S. R., WU, W. H., YIN, Y., and WU, W. H. The single-machine total tardiness problem with unequal release times and a linear deterioration. *Applied Mathematics and Computation*, 2013a, vol. 219, no. 20, p. 10401–10415.

WU, W. H., XU, J., WU, W. H., YIN, Y., CHENG, I. F., and WU, C. C. A tabu method for a two-agent single-machine scheduling with deterioration jobs.*Computers & Operations Research*, 2013b, vol. 40, no. 8, p. 2116–2127.

XU, K., FENG, Z., and JUN, K. A tabu-search algorithm for scheduling jobs with controllable processing times on a single machine to meet due-dates.*Computers & Operations Research*, 2010, vol. 37, no. 11, p. 1924-1938.

YALAOUI, F., and CHU, C. Parallel machine scheduling to minimize total tardiness. *International Journal of Production Economics*, 2002, vol. 76, no. 3, p. 265-279.

YANG, S. J. Parallel machines scheduling with simultaneous considerations of position-dependent deterioration effects and maintenance activities. *Journal of the Chinese Institute of Industrial Engineers*, 2011, vol. 28, no. 4, p. 270-280.

YANG, D. L., CHENG, T. C. E., YANG, S. J., and HSU, C. J. Unrelated parallel-machine scheduling with aging effects and multi-maintenance activities.*Computers & Operations Research*, 2012, vol. 39, no. 7, p. 1458-1464.

YANG, S. J. Unrelated parallel-machine scheduling with deterioration effects and deteriorating multi-maintenance activities for minimizing the total completion time.*Applied Mathematical Modelling*, 2013, vol. 37, no. 5, p. 2995–3005.

ZHANG Z., ZHENG, L., LI, N., WANG, W., ZHONG, S., and HU, K. Minimizing mean weighted tardiness in unrelated parallel machine scheduling with reinforcement learning. Computers & Operations Research, 2012, vol. 39, no. 7, pp. 1315-1324.

BIOGRAFÍAS

Alex Ruiz-Torres: Ruiz-Torres es Catedrático Asociado en la Facultad de Administración de Empresas, Universidad de Puerto Rico – Rio Piedras. Estudió la licenciatura, maestría y doctorado en ingeniería Industrial en Georgia Tech, Stanford, y PennState respectivamente. Ha recibido becas de la NASA, la Fundación Nacional de Ciencias de EEUU (NSF), y la Comisión Fulbright. Sus estudios de investigación se enfocan en modelos de cadenas de suministros, planificación de producción, y sistemas de decisión. Sus publicaciones han aparecido en revistas académicas internacionales incluyendo International Journal of Production Research, European Journal of Operational Research, International Journal of Production Economics, Computers and Operations Research, Journal of the Operational Research Society, Computers and Industrial Engineering, y OMEGA.

José H. Ablanedo Rosas: Ablanedo Rosas es Catedrático Asistente en la Escuela de Administración de Empresas de la Universidad de Texas en El Paso. El realizó sus estudios doctorales en la Universidad de Mississippi. Sus áreas de investigación son los modelos de la cadena de suministro, logística, planificación de producción, meta-heurísticas y la gerencia de la calidad. Sus trabajos se han publicado en revistas como European Journal of Operational Research, Computers and Operations Research, International Journal of Production Research, Expert Systems, OMEGA, e International Journal of Shipping and Transport Logistics.

Nelson Alomoto. Alomoto es Catedrático en la Facultad de Administración de Empresas, Escuela Politécnica Nacional localizada en Quito, Ecuador. Él tiene una licenciatura en Matemáticas y una Maestría en Ingeniería Industrial de la Escuela Politécnica Nacional. Sus áreas de investigación son la gerencia de los sistemas de producción, simulación de procesos, y la competitividad. El profesor Alomoto ha presentado sus investigaciones en conferencias internacionales y ha sido consultor para agencias gubernamentales y el sector privado en Ecuador.

Diana Jadan Avilés: Diana Jadan Avilés es Catedrática en las Carreras de Ingeniería Industrial e Ingeniería Química de la Facultad de Ciencias Químicas de la Universidad de Cuenca – Ecuador. La Ing. Jadan obtuvo su título como Ingeniera Industrial en la Universidad de Cuenca en el 2009 y su título de Máster en Ciencias en Sistemas de Producción y Logística en la "Ecole des Mines de Nantes" de Francia en el año 2012.