ISBN: 978-628-95207-4-3. ISSN: 2414-6390. Digital Object Identifier: https://dx.doi.org/10.18687/LACCEI2023.1.1.705

An Application of the Piecewise Linear Approximation Method to Solve the β-Robust Scheduling Problem in a Parallel Machine Environment

Miguel Fernández Pérez, PhD
Department of Engineering, Pontifical Catholic University of Peru, Lima 32, Peru, mfernandezp@pucp.edu.pe

Abstract*–In the industries, the presence of uncertainties in the duration of production tasks can lead to unsatisfactory production scheduling, in situations where variability is significant, and a deterministic approach is used. In this case, a stochastic or robust approach is more appropriate. In particular, this article deals with the β-robust scheduling problem in a parallel machine environment and considering the presence of uncertainty in the duration of the tasks. This problem consists of finding the execution order of a set of tasks on a set of machines, with the objective of maximizing the probability that the total flow time is less than a limit. The difficulty in solving this problem lies in its combinatorial, stochastic and non-linear nature of its formulation. To overcome this difficulty, an efficient mathematical model is built that makes use of the piecewise linear approximation method. The proposed model proves to be able to obtain the solution of the problem with precision and in a short computational time.

Keywords-- β-robust scheduling problem; mathematical model; piecewise linear approximation method; central limit theorem.

Una Aplicación del Método de Aproximación Lineal por Partes para la Resolución del Problema de Programación de Producción β-robust en un Ambiente de Máquinas en Paralelo

Miguel Fernández Pérez, PhD
Department of Engineering, Pontifical Catholic University of Peru, Lima 32, Peru, mfernandezp@pucp.edu.pe

***Resumen*– En las industrias, la presencia de incertezas en la duración de las tareas de producción puede llegar a generar una programación de la producción no satisfactoria, en situaciones donde la variabilidad es significativa y se utiliza un enfoque determinístico. En este caso, un enfoque estocástico o robusto es más apropiado. En particular, en este artículo se aborda el problema de programación de producción β-robust en un ambiente de máquinas en paralelo y considerando la presencia de incerteza en la duración de las tareas. Este problema consiste en encontrar el orden de ejecución de un conjunto de tareas en un conjunto de máquinas, con el objetivo de maximizar la probabilidad que el tiempo de flujo total sea menor a un límite. La dificultad en la resolución de este problema yace en su naturaleza combinatoria, estocástica y no lineal de su formulación. Para superar está dificultad, se construye un modelo matemático eficiente que hace uso del método de aproximación lineal por partes. El modelo propuesto demuestra ser capaz obtener la solución del problema con precisión y en un corto tiempo computacional.**

***Palabras clave*-- Programación de producción β -robust; modelo matemático; método de aproximación lineal por partes; teorema del límite central.**

I. Introducción

El concepto de programa β-robust en el ámbito de problemas de producción considerando la incertidumbre en la duración de las tareas fue introducido en la Referencia [1]. Un programa β-robust es aquel que tiene la máxima probabilidad de alcanzar un nivel de desempeño específico. Con frecuencia, el nivel de rendimiento representa la probabilidad de que el tiempo de flujo total (TFT) sea inferior a un límite. El TFT se define como la suma del tiempo de finalización de todas las tareas. En la Referencia [1], los autores formularon el problema de programación de producción β-robust para un entorno de una sola máquina; además, establecieron la complejidad del problema y desarrollaron enfoques de solución exactos y heurísticos.

En la Referencia [2] consideran el problema de programación de producción con una sola máquina y muestran cómo representar este problema como un modelo de restricciones. Los autores desarrollan tres modelos (primal, dual e híbrido) y muestran el efecto de las reglas de dominancia en el espacio de búsqueda.

Digital Object Identifier: (only for full papers, inserted by LACCEI).
ISSN, ISBN: (to be inserted by LACCEI).
DO NOT REMOVE

En la Referencia [3] se aborda este problema en un ambiente job shop, considerando que la duración de las tareas se ajusta a una variable aleatoria normal. Aquí, la función objetivo consiste en maximizar la probabilidad de que el makespan no exceda una fecha de vencimiento dada. Para resolver este problema, los autores desarrollaron un algoritmo Branch and Bound acompañado de un conjunto de reglas de dominancia.

En la Referencia [4], los autores propusieron un algoritmo Branch and Bound para resolver el problema en un ambiente de máquinas en paralelo. Además, se desarrollaron un límite inferior, tres límites superiores y dos reglas de dominancia para el algoritmo Branch and Bound.

En la Referencia [5], los autores estudiaron el problema de programación de producción en un ambiente de máquinas paralelo, considerando que la duración de las tareas se adecua a una variable aleatoria normal. El objetivo es maximizar la probabilidad de que todas las tareas se completen antes de una fecha de vencimiento común. Además, los autores desarrollaron un límite inferior y un límite superior para la función objetivo y un algoritmo eficiente Branch and Price.

II. Nomenclatura

La nomenclatura usada para definir los índices, parámetros y variables de decisión de los modelos matemáticos que se van a presentar en las próximas secciones se detallan a continuación:

Índices
- i: Índice de tareas, $i = \{1, 2, \ldots, J\}$
- m: Índice de máquinas, $m = \{1, 2, \ldots, N\}$
- t, h: Índice de tiempo, $t, h = \{1, 2, \ldots, T\}$
- k: Índice de nivel de prioridad, $k = \{1, 2, \ldots, K\}$
- b: Índice de partición de intervalo, $b = \{1, 2, \ldots, B\}$

Parámetros
- J: Número de tareas
- N: Número de máquinas
- T: Horizonte tiempo
- K: Máximo nivel de prioridad
- B: Número de puntos de corte del intervalo
- d_i: Duración de la tarea i

- μ_i: Valor esperado de la duración de la tarea i
- σ_i^2: Varianza de la duración de la tarea i
- S: Nivel de servicio al cliente

Variables
- $S_{imt} = 1$, si la máquina m comienza a realizar la tarea i en el tiempo t; 0, caso contrario.
- $L_{imk} = 1$, si la tarea i es realizado por la máquina m con un nivel de prioridad k; 0, caso contrario.
- $L'_{ik} = 1$, si la tarea i posee un nivel de prioridad k; 0, caso contrario.
- Y: Variable aleatoria normal.
- R: Desviación estándar del tiempo de flujo total.

III. Problema de programación de máquinas en paralelo

Considere un conjunto de J tareas a ser realizadas en un conjunto de N máquinas paralelas e idénticas. Cada máquina puede procesar como máximo una tarea a la vez y cada tarea debe ser realizada por una sola máquina. Cada tarea i está disponible para procesamiento en el tiempo cero y tiene asociado una duración d_i. No se permite la preferencia de tareas. De este modo, el problema de programación de máquinas en paralelo (PPMP) consiste en determinar la secuencia de procesamiento de las tareas con el objetivo de minimizar el TFT. Este problema es conocido como $P||\sum C_j$ y se caracteriza por pertenecer a la clase NP-hard [6].

A continuación, se presenta un modelo matemático para el PPMP, el cual usa variables indexadas en el tiempo. En la Referencia [7], los autores señalan que este tipo de modelos son útiles para problemas de gran porte.

$$\text{(PPMP-1) Min TFT} = \sum_{i=1}^{J}\sum_{m=1}^{N}\sum_{t=1}^{T}(t + d_i - 1)S_{imt} \quad (1)$$

Sujeto a:

$$\sum_{m=1}^{N}\sum_{t=1}^{T} S_{imt} = 1 \quad \forall i \quad (2)$$

$$\sum_{i=1}^{J}\sum_{h=t-d_i+1}^{t} S_{imh} \leq 1 \quad \forall m, t \quad (3)$$

$$S_{imt} \in \{0,1\}\ \forall i, m, t | 1 \leq t \leq T - d_i + 1 \quad (4)$$

Este modelo matemático es denotado como PPMP-1. En este modelo, la función objetivo (1) representa minimización del TFT. La restricción (2) asegura que el inicio de cada tarea ocurra solo una vez en una máquina y en un tiempo específico. La restricción (3) asegura que una máquina inicie como máximo una tarea en un tiempo específico y asegura que no haya interferencia entre las tareas que usan la misma máquina. La restricción (4) define el dominio de las variables de decisión y asegura que el inicio de los trabajos esté dentro del horizonte de tiempo.

En este artículo se propone un modelo matemático alternativo al PPMP-1. Se busca adaptar el cálculo del TFT, para este fin, se ilustra el cálculo del TFT para un caso con 4 tareas y una máquina. Sea la duración d_i de la tarea $i = 1, 2, 3, 4$, y la secuencia de tareas a ser realizadas $\{J_1, J_2, J_3, J_4\}$, entonces, el TFT es dado por:

$$\left(d_{J_1}\right) + \left(d_{J_1} + d_{J_2}\right) + \left(d_{J_1} + d_{J_2} + d_{J_3}\right) + (d_{J_1} + d_{J_2} + d_{J_3} + d_{J_4})$$

Esta expresión aún puede ser reducida como:

$$(1)d_{J_4} + (2)d_{J_3} + (3)d_{J_2} + (4)d_{J_1}$$

El valor que multiplica a cada duración puede ser entendido como el peso relativo de cada tarea en la secuencia. De aquí para adelante cada peso relativo será llamado de nivel de prioridad. Para este ejemplo, los niveles de prioridad son 1, 2, 3, 4. Note que la tarea con nivel de prioridad 4 tendrá mayor prioridad en la secuencia que las tareas con niveles 1, 2, 3; la tarea con nivel de prioridad 3 tendrá mayor prioridad en la secuencia que las tareas con niveles 1, 2; y así sucesivamente. Note que el nivel de prioridad es inverso al orden ejecución de las tareas; y esta misma deducción puede ser aplicada para problemas con más de una máquina. De esta manera, se puede formular un modelo matemático alternativo (PPMP-2) de la siguiente forma:

$$\text{(PPMP-2) Min TFT} = \sum_{i=1}^{J}\sum_{m=1}^{N}\sum_{k=1}^{K} k d_i L_{imk} \quad (5)$$

Sujeto a:

$$\sum_{m=1}^{N}\sum_{k=1}^{K} L_{imk} = 1 \quad \forall i \quad (6)$$

$$\sum_{i=1}^{J} L_{imk} \leq 1 \quad \forall m, k \quad (7)$$

$$L_{imk} \in \{0,1\}\ \forall i, m, k \quad (8)$$

En este modelo matemático, la función objetivo (5) representa la minimización del TFT; y se demuestra que el valor obtenido de la función objetivo en (1) y (5) es el mismo. La restricción (6) asegura que cada tarea seleccione una máquina y un nivel de prioridad. La restricción (7) asegura que una máquina, con un nivel específico de prioridad, se le asigne una tarea. La restricción (8) define el dominio de las variables de decisión.

PPMP-2 puede ser simplificado reemplazando la variable L_{imk} por L'_{ik}, y aplicando una sumatoria sobre el índice m en la restricción (7). El modelo simplificado (PPMP-3) es formulado como:

$$(\text{PPMP-3})\ \text{Min TFT} = \sum_{i=1}^{J}\sum_{k=1}^{K} k\, d_i L'_{ik} \tag{9}$$

Sujeto a:

$$\sum_{k=1}^{K} L'_{ik} = 1\ \forall i \tag{10}$$

$$\sum_{i=1}^{J} L'_{ik} \leq N \quad \forall k \tag{11}$$

$$L'_{ik} \in \{0,1\}\ \forall i,k \tag{12}$$

En este modelo matemático, la función objetivo (9) representa la minimización del TFT. La restricción (10) asegura que a cada tarea se le asigne un nivel de prioridad. La restricción (11) asegura que un dado nivel de prioridad se le asigne a lo mucho a N tareas simultáneamente, ya que las máquinas se encuentran en paralelo. La restricción (12) define el dominio de las variables de decisión.

PPMP-3 será utilizado como base para formular en la próxima sección el problema de programación de producción β-robust en un ambiente de máquinas en paralelo. PPMP-3 es adecuado debido a que el parámetro incierto, que es la duración de las tareas, se encuentra en la función objetivo, en consecuencia, es apropiado para aplicar funciones estadísticas; note que el parámetro incierto no tendrá impacto en las restricciones del modelo.

IV. Problema de programación β -robust

El problema de programación β-robust es una extensión del PPMP. La principal diferencia es la incertidumbre presente en duración de cada tarea. En este problema, cada tarea i está asociada a una duración d_i , que es una variable aleatoria independiente normalmente distribuida con media μ_i y varianza σ_i^2, el cual es expresado como $d_i \sim N(\mu_i, \sigma_i^2)$. Así, el problema de programación β-robust consiste en encontrar la secuencia que maximice la probabilidad que el TFT no exceda un límite predeterminado S (nivel de servicio al cliente). Este problema es conocido como $P|\beta - \text{RSP}|TFT \leq S$ [4].

Teniendo en consideración el teorema del límite central, el TFT determinado en la ecuación (9) se puede aproximar por una distribución normal de la siguiente manera:

$$TFT \sim N\Big(\sum_{i=1}^{J}\sum_{k=1}^{K} k\, \mu_i L'_{ik}\ ,\ \sum_{i=1}^{J}\sum_{k=1}^{K} k^2\, \sigma_i^2 L'_{ik}\Big)$$

De esta forma, el problema de programación β-robust es formulado como:

$(\beta - \text{RSP1})$

$$\text{Max}\ Y = \left(S - \sum_{i=1}^{J}\sum_{k=1}^{K} k\, \mu_i L'_{ik}\right) \Bigg/ \sqrt{\sum_{i=1}^{J}\sum_{k=1}^{K} k^2\, \sigma_i^2 L'_{ik}} \tag{13}$$

Sujeto a:

$(10) - (12)$

En este modelo matemático, la función objetivo (13) maximiza el valor de una variable aleatoria normal, lo que implica la maximización de la probabilidad que el TFT sea menor a S.

Definiendo R como la desviación estándar del TFT se tiene:

$$(\beta - \text{RSP2})\ \text{Max}\ Y = \left(S - \sum_{i=1}^{J}\sum_{k=1}^{K} k\, \mu_i L'_{ik}\right) \Big/ R \tag{14}$$

Sujeto a:

$(10) - (12)$

$$R^2 = \sum_{i=1}^{J}\sum_{k=1}^{K} k^2\, \sigma_i^2 L'_{ik} \tag{15}$$

$$R \geq 1 \tag{16}$$

En este artículo, por conveniencia se asume que, en todos los casos, la desviación estándar del TFT siempre es mayor o igual a uno. Note que este modelo matemático es no lineal, una forma de resolverlo es aplicar técnicas de linealización. En primer lugar, se va a introducir las siguientes variables:

$D = 1/R$, con $0 < D \leq 1$

$G_{ik} = L'_{ik} D \quad \forall i,k$

Reemplazando estas variables en el modelo matemático anterior se tiene:

$$(\beta - \text{RSP3})\ \text{Max}\ Y = SD - \sum_{i=1}^{J}\sum_{k=1}^{K} k\, \mu_i G_{ik} \tag{17}$$

Sujeto a:

$$\sum_{k=1}^{K} G_{ik} = D\ \ \forall i \tag{18}$$

$$G_{ik} \leq L'_{ik}\ \ \forall i,k \tag{19}$$

$$G_{ik} \leq D\ \ \forall i,k \tag{20}$$

$$G_{ik} \geq D - (1 - L'_{ik})\ \ \forall i,k \tag{21}$$

$$\sum_{i=1}^{J} G_{ik} \leq ND \quad \forall k \tag{22}$$

$$R = \sum_{i=1}^{J}\sum_{k=1}^{K} k^2\, \sigma_i^2 G_{ik} \tag{23}$$

$$D = \frac{1}{R} \tag{24}$$

$$L'_{ik} \in \{0,1\}\ \ \forall i,k \tag{25}$$

$$G_{ik} \geq 0\ \ \forall i,k \tag{26}$$

$$R \geq 1 \tag{27}$$

$$0 < D \leq 1 \tag{28}$$

En este modelo matemático, la restricción (24) es una expresión no lineal. Con la finalidad de resolver el problema de programación β-robust de manera eficiente, en este artículo es utilizado el método de aproximación lineal por partes para determinar la varianza del TFT. Este método es útil para transformar funciones no lineares en una suma compuesta por términos lineales. La aplicación del método requiere que sea preestablecido un límite inferior y un límite superior de la función no linear. Estos límites apenas precisan ser confiables para realizar la aproximación deseada y no es necesario desarrollar métodos sofisticados para determinarlos.

A. Límite inferior de la varianza del TFT

En este artículo, el límite inferior de la varianza del TFT (R_{LB}^2) es determinado mediante un modelo de optimización:

$$R_{LB}^2 = \text{Min} \sum_{i=1}^{J} \sum_{k=1}^{K} k^2 \, \sigma_i^2 L'_{ik} \tag{29}$$

Sujeto a:

$(10) - (12)$

En este modelo matemático, la función objetivo minimiza la varianza del TFT.

B. Límite superior de la varianza del TFT

Un límite superior para la varianza del TFT (R_{UB}^2) puede ser encontrado construyendo una secuencia de tareas aplicando el método del shortest expected processing time (SEPT). Este método asigna las tareas a las máquinas de manera cíclica según el orden decreciente a la duración de las tareas. Este límite es determinado como $R_{UB}^2 = \text{Var}[TFT_{SEPT}]$.

C. Aproximación lineal por partes

La aproximación lineal por partes es un método para generar un modelo de programación lineal entera a partir de un modelo de programación no lineal. Este método es útil cuando se encaran problemas que no pueden ser resueltos empleando solvers comerciales.

La aproximación lineal por partes busca transformar una función no lineal en una suma de funciones lineales mediante la definición de puntos conocidos como puntos de corte que dividen un dominio preestablecido de una cierta variable de decisión. Cualquier punto entre dos puntos de corte y su correspondiente valor de su función es determinada por una suma ponderada.

En este artículo, sobre el modelo $\beta - \text{RSP3}$ será aplicado el método de aproximación lineal por partes para transformar la restricción fraccional (24) en una restricción lineal. Para este fin, la variable R que representa la desviación estándar del TFT será dividido en un conjunto de puntos de corte X_b, $b = 1, \dots, B$, dentro del rango $[R_{LB}, R_{UB}]$:

$$X_b = R_{LB} + (b-1)(R_{UB} - R_{LB})/(B-1)$$

La restricción (24) puede ser sustituida por tres ecuaciones, las cuales se muestran abajo:

$$\tilde{R} = \sum_{b=1}^{B} X_b \lambda_b \tag{30}$$

$$\widetilde{D} = \sum_{b=1}^{B} \frac{\lambda_b}{X_b} \tag{31}$$

$$\sum_{b=1}^{B} \lambda_b = 1 \tag{32}$$

La variable λ_b representa el peso assignado al parámetro X_b en cada partición b. La restricción (30) determina la desviación estándar por medio de una suma ponderada de todos los puntos de corte. La restricción (31) determina $1/R$ por medio de una suma ponderada de la inversa de todos los puntos de corte. La restricción (32) señala que la suma de los pesos debe ser igual a uno. Además, se debe garantizar que a lo mucho dos pesos adyacentes sean mayores que cero, y el resto de los pesos iguales a cero. Este requerimiento ha sido configurado dentro de un módulo propio del software de optimización. La precisión de la aproximación de $1/R$ depende de B, esto es, la cantidad de puntos de corte generados.

A continuación, se presenta el modelo matemático propuesto, que incorpora las restricciones producidas por la aproximación lineal por partes, para resolver el problema de programación β-robust:

$$(\beta - \text{RSP4}) \ \text{Max} \, Y = S\widetilde{D} - \sum_{i=1}^{J} \sum_{k=1}^{K} k \, \mu_i G_{ik} \tag{33}$$

Sujeto a:

$(19) - (21), (25), (26), (30) - (32)$

$$\sum_{k=1}^{K} G_{ik} = \widetilde{D} \ \ \forall i \tag{34}$$

$$\sum_{i=1}^{J} G_{ik} \leq N\widetilde{D} \ \ \forall k \tag{35}$$

$$\tilde{R} = \sum_{i=1}^{J} \sum_{k=1}^{K} k^2 \, \sigma_i^2 G_{ik} \tag{36}$$

$$\tilde{R} \geq 1 \tag{37}$$

$$0 < \widetilde{D} \leq 1 \tag{38}$$

$$\lambda_b \geq 0 \ \ \forall b \tag{39}$$

El modelo $\beta - \text{RSP4}$ será utilizado para realizar los experimentos computacionales en la próxima sección.

V. EXPERIMENTOS COMPUTACIONES

Para evaluar el desempeño del modelo matemático propuesto ($\beta - \text{RSP4}$) para resolver el problema de programación β-robust se utilizaron varias instancias. Los parámetros del problema se definieron siguiendo las referencias [1], [2] y [8]. La media y la varianza de la duración de cada tarea i han sido generadas utilizando una distribución uniforme:

$$\mu_i \sim U(10{,}50\delta_1)$$
$$\sigma_i^2 \sim U(0, \frac{1}{9}\mu_i^2\delta_2)$$

con $\delta_1 = \{0.4{,}0.7{,}1.0\}$ y $\delta_2 = \{0.4{,}0.7{,}1.0\}$. Los valores de μ_i y σ_i^2 que son generados se redondean a números enteros. Los valores de z por nivel de servicio requerido han sido seleccionados con probabilidades de 0.85, 0.95 y 0.99. El nivel de servicio es determinado usando la media del TFT, la varianza obtenida por el método SEPT y el valor de z:

$$S = E[TFT_{SEPT}] + z\sqrt{Var(TFT_{SEPT})}$$

Las instancias utilizadas consisten en una combinación de δ_1, δ_2 y el nivel de servicio. Para cada combinación son generadas 10 instancias.

Al parámetro B (número de puntos de corte) se le asigno un valor de 1000.

También fueron definidos dos indicadores de eficiencia para cada solución obtenida a partir del modelo propuesto. El primer indicador de Eficiencia (E_1) compara la probabilidad de que el programa β-robust alcance el nivel de servicio requerido, con la probabilidad de que el programa SEPT logre el mismo nivel de servicio. El segundo indicador de eficiencia (E_2) compara el riesgo de que el programa β-robust falle en alcanzar el nivel de servicio requerido, con el riesgo de que el programa SEPT falle en alcanzar el mismo nivel de servicio [2].

$$E_1 = \frac{\Pr(TFT \leq S) - \Pr(TFT_{SEPT} \leq S)}{\Pr(TFT_{SEPT} \leq S)}$$
$$E_2 = \frac{(1 - \Pr(TFT_{SEPT} \leq S)) - (1 - \Pr(TFT \leq S))}{(1 - \Pr(TFT \leq S))}$$

El modelo matemático propuesto ha sido implementado en el software AIMMS 3.14, utilizando el solver CPLEX 12.4 y se ejecutó en una computadora equipada con un procesador Intel Core i3-2310M con 2.10 GHz y 4 GB de RAM.

A. Instancias con una sola máquina

En esta sección son presentados los resultados de los experimentos computacionales con 10, 15 y 20 tareas. El conjunto de problemas está compuesto por 810 instancias. La Tabla I muestra los valores medios del tiempo de ejecución; la desviación estándar del error ($\tilde{R} - R$), que resulta de la aplicación del método de la aproximación lineal por partes; y los indicadores de eficiencia para los diferentes valores de z y del número de tareas.

TABLA I.
RESULTADOS COMPUTACIONALES PARA INSTANCIAS CON UNA SOLA MÁQUINA

z	Número de tareas	Tiempo (s)	$\tilde{R} - R$	E_1 (%)	E_2 (%)
1.04	10	0.12	1.8E-07	0.59	3.65
	15	0.22	2.7E-07	0.51	3.06
	20	0.36	5.5E-07	0.52	3.09
1.65	10	0.11	1.6E-07	0.50	12.30
	15	0.22	2.9E-07	0.46	10.28
	20	0.37	4.1E-07	0.39	8.31
2.33	10	0.12	1.6E-07	0.22	37.69
	15	0.32	4.1E-07	0.21	30.47
	20	0.38	4.8E-07	0.15	19.89

La Tabla I muestra que el modelo matemático propuesto puede resolver eficientemente el problema en estudio. Note que el tiempo computacional requerido para resolver cualquier instancia es menor a 0.4 s y el error de la aproximación lineal por partes es cercano a cero. Para todas las instancias, el programa β-robust aumenta la probabilidad de alcanzar el nivel de servicio por encima del programa SEPT (E_1) en un 0.4% en promedio. En las instancias con valor de z de 1.04, 1.65 y 2.33, el programa β-robust reduce el riesgo sobre el programa SEPT (E_2) en 3.27%, 10.3% y 29.35% en promedio, respectivamente.

B. Instancias con varias máquinas en paralelo

En esta sección son presentados los resultados de los experimentos computaciones con 30, 35, 40 y 45 tareas y con 3, 4 y 5 máquinas. El conjunto de problemas está compuesto por 3240 instancias. La Tabla II muestra los valores medios del tiempo de ejecución; la desviación estándar del error ($\tilde{R} - R$), que resulta de la aplicación del método de la aproximación lineal por partes; y los indicadores de eficiencia para los diferentes valores de z, el número de tareas y el número de máquinas.

TABLA II.
RESULTADOS COMPUTACIONALES PARA INSTANCIAS CON VARIAS MÁQUINAS EN PARALELO

z	Número de tareas	Número de máquinas	Tiempo (s)	$\tilde{R} - R$	E_1 (%)	E_2 (%)
1.04	30	3	0.42	3.1E-07	0.47	2.79
		4	0.37	2.1E-07	0.45	2.67
		5	0.32	1.2E-07	0.40	2.38
	35	3	0.61	4.1E-07	0.47	2.77
		4	0.51	2.4E-07	0.44	2.64
		5	0.43	1.8E-07	0.47	2.79
	40	3	0.82	7.2E-07	0.50	2.96
		4	0.66	2.6E-07	0.45	2.66
		5	0.58	2.6E-07	0.46	2.75
	45	3	1.18	5.6E-07	0.43	2.56
		4	0.88	4.9E-07	0.43	2.57
		5	0.75	2.7E-07	0.45	2.64
1.65	30	3	0.45	2.7E-07	0.35	7.31
		4	0.39	2.1E-07	0.36	7.81
		5	0.33	1.7E-07	0.34	7.28
	35	3	0.61	4.1E-07	0.31	6.44
		4	0.54	2.7E-07	0.33	6.91
		5	0.44	2.0E-07	0.31	6.46
	40	3	0.81	4.0E-07	0.32	6.62
		4	0.65	3.1E-07	0.33	7.03
		5	0.58	2.6E-07	0.35	7.54
	45	3	1.11	5.2E-07	0.33	6.86
		4	0.90	5.6E-07	0.31	6.57
		5	0.78	3.4E-07	0.31	6.49
2.33	30	3	0.45	3.7E-07	0.15	18.31
		4	0.39	2.2E-07	0.16	20.01
		5	0.33	1.9E-07	0.14	17.90
	35	3	0.62	4.0E-07	0.14	16.92
		4	0.51	4.0E-07	0.13	15.97
		5	0.47	2.1E-07	0.13	15.71
	40	3	0.84	4.2E-07	0.14	17.30
		4	0.68	3.2E-07	0.14	17.06
		5	0.61	2.4E-07	0.13	16.10
	45	3	1.18	5.0E-07	0.14	16.80
		4	0.91	3.9E-07	0.14	16.34
		5	0.77	3.4E-07	0.13	15.47

La Tabla II muestra que el modelo matemático propuesto también es útil para resolver el problema en estudio para diferente número de tareas y máquinas. El tiempo computacional requerido para resolver cualquier instancia es menor a 1.2 s y el error de la aproximación lineal por partes es cercano a cero. Para todas las instancias, el programa β-robusto aumenta la probabilidad de alcanzar el nivel de servicio por encima del programa SEPT (E_1) en un 0.31 % en promedio. En las instancias con valor z de 1.04, 1.65 y 2.33, el programa β-robust reduce el riesgo sobre el programa SEPT (E_2) en 2.68%, 6.94% y 16.99% en promedio, respectivamente.

En las tablas I and II, se percibe que el programa SEPT no presenta una diferencia significativa con programa β-robust cuando se compara el indicador de eficiencia E_1; sin embargo, en relación con el indicador de eficiencia E_2, esta diferencia se incrementa conforme aumente el valor de z.

VI. CONCLUSIONES

En este artículo, se presenta un nuevo modelo matemático para resolver el problema de programación de producción β-robust. Este problema es una extensión del problema de programación de máquinas en paralelo, en el cual se considera la incerteza en la duración las tareas. El modelo propuesto está basado en un modelo presentado en la Referencia [7]. Para llegar al modelo propuesto, en primer lugar, se desarrolla un modelo que usa variables indexadas en el tiempo; posteriormente se realizan adaptaciones que dejan a las restricciones libres de la influencia de la incerteza de la duración de las tareas; sin embargo, obtiene una función objetivo no lineal. Por último, se aplica el método de la aproximación lineal por partes con la finalidad de linealizar la expresión de la desviación estándar del tiempo de flujo total.

Los experimentos computacionales demuestran que el modelo propuesto resolvió de manera satisfactoria todas las instancias generadas y en un tiempo computacional reducido; se tiene que resaltar que se obtiene una excelente precisión por la aplicación del método de la aproximación lineal por partes.

El modelo propuesto es adecuado en sistemas de producción, en aquellas situaciones en que un enfoque determinístico no es el apropiado por generar soluciones ambiguas. La solución obtenida por el modelo propuesto produce una secuencia robusta de tareas a ser ejecutadas en las máquinas.

REFERENCES

[1] R. L. Daniels, J. E. Carrillo, "β-Robust scheduling for single-machine systems with uncertain processing times," *IIE transactions*, vol. 29, no. 11, pp. 977-985, 1997.

[2] C. W. Wu, K. N. Brown, J. C. Beck, "Scheduling with uncertain durations: Modeling β-robust scheduling with constraints," *Computers & Operations Research*, vol. 36, no. 8, pp. 2348-2356, 2009.

[3] S. M. Khatamia, M. Ranjbara, M. Davari, "Maximizing service level in a β-robust job shop scheduling model," *Journal of Industrial and Systems Engineering*, vol. 8, no. 4, pp. 59-71, 2015.

[4] S. Alimoradi, M. Hematian, G. Moslehi, "Robust scheduling of parallel machines considering total flow time," *Computers & Industrial Engineering*, vol. 93, pp. 152-161, 2016.

[5] A. Novak, P. Sucha, M. Novotny, R. Stec, Z. Hanzalek, "Scheduling jobs with normally distributed processing times on parallel machines," *European Journal of Operational Research*, vol. 297, no. 2, pp. 422-441, 2022.

[6] M. Pinedo, *Scheduling: theory, algorithms, and systems*, Prentice Hall: New Jersey, 2008.

[7] M. Fernández, F. Oliveira, S. Hamacher, "A new mathematical model for the workover rig scheduling problem," *Pesquisa Operacional*, vol. 36, pp. 241-257, 2016.

[8] M. Ranjbar, M. Davari, R. Leus, "Two branch-and-bound algorithms for the robust parallel machine scheduling problem," *Computers & Operations Research*, vol. 39, no. 7, pp. 1652-1660, 2012.